Swarm Intelligence Course (INFO-H-414)
Exam Second Session

July 3, 2018

1 Modality

The exam is divided in two parts:

Project You are asked to provide the required deliverables and to present your project in a 7-minute talk, followed by 5 minutes of questions. This will account for up to 10 points of your final grade.

Questions You will be asked a number of questions concerning the entire course material. This will account for up to 10 points of your final grade.

To pass the exam one must collect at least 5 points for each part of the exam.

1.1 Timeline

• The date of the exam is August 21th, 2018.
• The project submission deadline is August 15th, 2018 at 23.59.
• Each 12 hours of delay on the submission will entail a penalty of 1 point on the final evaluation of the project.

1.2 Project Deliverables

For the project, you will have to provide:

• Your code in digital format (i.e., text files), so we can test it. Send it by e-mail to the people responsible for the project (see contacts at the end of the document).

• A short document (6-8 pages) written in English that describes your work. You have to explain both the idea and the implementation of your solution, and present the results you obtained.

• On the day of the oral presentation, you are expected to show slides and come with your own laptop. If you happen to not have a laptop, send us a PDF version of your slides before the day of the exam. You will also be asked to show a live demo of your solution or to explain your code.
2 Project

2.1 General Remarks

Apply what you learnt. It is mostly important that you stick to the principles of swarm intelligence: simple, local interactions, no global communication, no global information.

Avoid complexity. If your solution gets too complex, it is because it is the wrong one.

Honesty pays off. If you find the solution in a paper or in a website, cite it and say why you used that idea and how (it is not a negative thing implementing solution existing in the literature).

Cooperation is forbidden. Always remember that this is an individual work.

Analyse your solution. Provide a good analysis of the results of your solution (generate data, plot graphs, report statistical measures). Do not limit your project to the implementation of a nice solution. The result analysis is as important as the code implementation.

The project counts for 50% of your final grade. The basic precondition for you to succeed in the project is that it must work. If it does not, the project won't be considered sufficient. In addition, code must be understandable — comments are mandatory.

The document is very important too. We will evaluate the quality of your text, its clarity, its completeness and its soundness. References to existing methods are considered a plus — honesty does pay off! More specifically, the document is good if it contains all the information needed to reproduce your work without having seen the code and a good and complete analysis of the results.

The oral presentation is also very important. In contrast to the document, a good talk deals with ideas and leaves the technical details out. Be sure that it fits in the 7-minute slot.

2.2 Swarm Robotics Project:

Task Sequencing with Ambient Cues

The activity of food search and retrieval is commonly referred to as foraging. In swarm robotics, foraging is a commonly used task to compare different algorithms for exploration (what is the best way to discover interesting places in the environment?), division of labor (who should explore? for how long?), etc. In the most general setting, food items are scattered in an environment at locations unknown to the robots and the robots need to explore the environment, find the food, and take it to the nest. Aggregation is another task commonly studied in swarm robotics. In this task, the swarm have to aggregate in order to gather the maximum number of robots in an specific place of the environment.
In this project we consider a mission comprising a sequential execution of the two aforementioned tasks. An environmental signal will tell the robots to switch from one task to the other. Indeed, during the first phase, the swarm has to first forage a source of food. Then, in the second phase, the robots must aggregate on the source of food as fast as possible. The students are asked to provide and study the performance of a responsive control software.

2.2.1 Problem definition

The robot swarm operates in a dodecagonal arena composed of a target source and a nest. The nest is represented by a black rectangle drawn on the floor. The source is represented by a gray circle drawn on the floor and the robots can perceive it through the ground sensor. The walls of the arena display LEDs that can be either blue or green and the color will represent the task that should be performed by the robots. The colors in the walls can be perceived using the omnidirectional camera. Finally, a light is placed above the nest. The robots can perceive it with the light sensor and it can be used to help the robots in the exploration of the environment.

During foraging phase, the walls of the arena are displaying a BLUE color. During the aggregation phase, the walls are GREEN. The duration of both phases is fixed: 800 s (8000 time-steps) for the foraging phase, and 200 s (2000 time-steps) for the aggregation phase. Each experiment is automatically terminated after 1000 s (10000 time-steps).

Goal     The goal of the robot swarm is to maximize the foraging objective function — that is, the number of items collected at the nest — and to minimize the aggregation objective function — that is, the number of robots outside the aggregation area. The overall evaluation will consider the performance of the swarm in both tasks.

Swarm composition     The swarm comprises 30 homogeneous robots. The robots are equipped with the following sensors and actuators:

- colored_blob_omnidirectional_camera to detect the walls LEDs;
- light to perceive the light above the nest;
- motor_ground to sense the color of the ground;
- proximity to avoid obstacles;
- wheels to explore the environment;
- leds to display information.
Additional Remarks

• ARGoS has been configured so that the robots collect items on sources and drop them at the nest automatically. A robot can only carry one item at a time.

• The position of the source can vary. See Figure 1.

• Your solution should be flexible: although the duration of both phases is fixed, your robot swarm should be able to adapt to different durations of both phases.

2.2.2 Goals

The goal of this project is to design, implement and test a robot swarm that firstly maximizes the number of items collected at the nest, and secondly it minimizes the time spend out of the source when the environment indicates the transition.

ARGoS is configured to automatically dump data on a file whose name can be changed in the .argos experiment configuration. This file is composed of a table containing three columns:

• The current step
• The number of items collected
• The score of the aggregation task (i.e. the sum of the timesteps for each robot outside of the source)
A complete analysis must be performed to evaluate the control software implemented. In particular, to present statistically meaningful results, we suggest you to execute and collect the results over at least 30 runs of each controller. The 30 runs of a controller should differ from each other for the random seed specified in the .argos experiment. In your report, you will include a table containing the following columns:

- Random seed
- Final number of items collected
- Final score of the aggregation task

Beside these two tables, quantitative numerical measures of the performance of your solution must be produced. Specifically, you should produce plots that show the trend over time of the controller implemented—e.g. number of items collected during the foraging phase, number of robots out of the source during the aggregation phase. On the basis of these measures, you should discuss the results and draw the appropriate conclusions.

Be aware that, for the project evaluation, the analysis is as important as the implementation. Make sure that the information provided in the report is meaningful, clearly written and complete. Any meaningful additional content will be rewarded.

Setting up the code

- Clone the argos3-arena repository into your $HOME directory

  $ git clone --branch h414 https://github.com/demiurge-project/argos3-arena.git

- Compile the code

  $ cd argos3-arena
  $ mkdir build
  $ cd build
  $ cmake ../src
  $ make


- Unpack the archive into your $HOME directory and compile the code

  $ tar -xzf SR_Project_H414.tar.gz
  $ cd SR_Project_H414
  $ mkdir build
  $ cd build
  $ cmake ../src
  $ make
• Set the environment variable ARGOS_PLUGIN_PATH to the full path in which the build/ directory is located:

$ export ARGOS_PLUGIN_PATH=$HOME/SR_Project_H414/build/

You can also put this line into your $HOME/.bashrc file, so it will be automatically executed every time you open a console.

• Run the experiment to check that everything is OK:

  $ cd $HOME/SR_Project_H414 # Enter the directory
  $ argos3 -c foraging.argos # Run the experiment

If the usual ARGoS GUI appears, you’re ready to go.

### 2.3 Setting up the experiment

**Switching the visualization on and off.** The experiment configuration file allows you to launch ARGoS both with and without visualization. When you launch ARGoS with the visualization, you can program the robots interactively exactly like you did during the course. Launching ARGoS without the visualization allows you to run multiple repetitions of an experiment automatically, e.g., through a script. By default, the script launches ARGoS in interactive mode. To switch the visualization off, just substitute the visualization section with:

```xml
<visualization />
```

, or, equivalently, comment out the entire qt-opengl section.

**Loading a script at init time.** When you launch ARGoS without visualization, you cannot use the GUI to set the running script. However, you can modify the XML configuration file to load automatically a script for you. At line 50 of foraging.argos you’ll see that the Lua controller has an empty section `<params />`. An example of how to set the script is at line 53 of the same file. Just comment line 50, uncomment line 53 and set the script attribute to the file name of your script.

**Changing the random seed.** When you want to run multiple repetitions of an experiment, it is necessary to change the random seed every time. To change the random seed, set the value at line 11 of foraging.argos, attribute random_seed.

**Changing the output file name.** As explained above, ARGoS automatically dumps data to a file as the experiment goes. To set the name of this file, set a new value for the attribute output at line 17 of foraging.argos.

**Making ARGoS run faster.** Sometimes ARGoS is a little slow, especially when many robots and many sensors are being simulated. You can make ARGoS go faster by setting the attribute threads at line 9 of foraging.argos. Experiment with the values, because the best setting depends on your computer.

#### 2.3.1 Deliverables

The final deliverables must include source code and documentation:
Code: The Lua scripts that you developed, well-commented and well-structured.

Documentation: A report of 6-8 pages structured as follows:

- Main idea of your approach.
- Structure of your solution (the state machine).
- Analysis and comparisons of the results.

3 Contacts

Marco Dorigo  mdorigo@ulb.ac.be for general questions
Mauro Birattari  mbiro@ulb.ac.be for general questions
David Garzon Ramos  dgarzonr@ulb.ac.be for swarm robotics
Antoine Ligot  aligot@ulb.ac.be for swarm robotics