Designing control software for robot swarms

Darko Bozhinoski
IRIDIA, Université Libre de Bruxelles

Mauro Birattari
IRIDIA, Université Libre de Bruxelles
Roadmap

- Robot Swarm
- Automatic Design Approach
- Research Lines
  - Defining formal abstractions
  - Defining benchmarks for evaluation
- Conclusions
Robot Swarm
Robot Swarm (2)

• Robot swarm operates in a self-organized and distributed manner

• Swarm collective behavior
  – Robot-robot interaction
  – Robot-environment interaction
Swarm Robotics vs. Decentralized Multi-Robot Systems

- **[Similarities]**
  - there is no leader
  - coordination is obtained via interaction between the individual robots

- **[Differences]**
  - Emergent behavior
  - Huge tiny robots, limited capabilities, limited resources
Swarm robotics (Scope)

• Environment characteristics:
  – High risk of individual robots failing or getting lost
  – Supporting infrastructures are hard or impossible to set up
  – Communication range is limited

• System Properties:
  – Fault-tolerance
  – Scalability
  – Flexibility
Challenges in swarm robotics

• Individual/Swarm dichotomy
Automatic Design Approach

Design Problem: Swarm level requirements

transformed

OPTIMIZATION Problem

Objective function
Optimization algorithm

SOLUTION: Control software for an individual robot
Automatic Design Approach (2)

- Evolutionary swarm robotics
  - control software of each individual robot is a neural network
  - evolutionary algorithm optimizes a mission-specific objective function
  - answer scientific questions related to the plausibility of biological models or the justification of animal behaviors in evolutionary terms
Automatic Design Approach (3)

- Evolutionary swarm robotics
  - control software of each individual robot is a neural network
  - evolutionary algorithm optimizes a mission-specific objective function
  - answer scientific questions related to the plausibility of biological models or the justification of animal behaviors in evolutionary terms

Drawback: Tailored to a specific application scenario without the possibility for generalization of the solution.
Research Lines
Defining formal abstractions

- Formal models of the environment and the system
- Formal language for specifying the requirements and constraints of a swarm
  - both functional and non-functional requirements

Objective:

A swarm designer should be able:

i. to specify the attributes of the robots
ii. Mission goals
iii. System and environment constraints
Defining benchmarks for evaluation

- Research Questions:
  - Which automatic method is the best under which conditions?
  - How general is method X?
  - How well does method X perform on different missions?
Defining benchmarks for evaluation (2)

- Automatic design methods are typically tested on a single mission by authors who introduced them.
- Compare different automatic design methods under the same conditions.
  - preventing experimenter bias

Objective:

**Automatic design method** is tested on **multiple missions** without undergoing any ad hoc, manual, per-mission modification.
Defining benchmarks for evaluation (3)

- Analysis on large set of different missions which are representative subset of the domain.
- Benchmark missions for the evaluation of different automatic design methods

**Objective:**

Software engineers develop principles and tools to automatically generate **benchmark missions** within a class of missions.
Conclusions

- A well-established software engineering discipline can make a contribution in:
  
  i. defining formal abstractions that could be reused in different missions and projects and
  
  ii. identifying and defining benchmarks that are important to be considered when designing and evaluating swarm control software.
Contributions we expect:

- Development of a classification framework that will increase reusability of automatic design methods;
- Coherent development of automatic design methods
- Tighter integration between research activities of different research groups.