Outline

- **Problem**
  - Issues with Single Robots
  - Proposition: Multiple Robots
  - Issues with Swarms of Robots

- **Biological inspiration**
  - Self-organization
  - Mechanisms

- **Swarm Intelligence**

- **Swarm robotic**
  - Hand designed behaviours
  - Behavioural architectures
  - Learning in Swarm Robotic
Single robots - Curiosity

Particularities

- One mission: explore mars
- Large range of tools
  - Large size and important weight
  - Complex landing
  - Costly mission ($ 2.5 billion)
Single robots - Roomba

Particularities

- One mission: clean your house
- Restricted set of tools
  - Middle-sized object for a house (35cm x 9cm)
  - Mass produced
  - Relatively expensive for a household ($ 700)
## Single robots - pros and cons

<table>
<thead>
<tr>
<th>Pros</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tailored for a specific task</td>
</tr>
<tr>
<td>Only interaction with environment to plan</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single point of failure</td>
</tr>
<tr>
<td>One robot for multiple tasks can become complex</td>
</tr>
<tr>
<td>and therefore expensive</td>
</tr>
</tbody>
</table>
Multiple robots

Idea

- Many individuals
- Simple sensing and computing
- Global emergent behaviour: self-organized process

Properties

- Scalability of behaviours:
  - Multiple simple robots realize tasks in parallel
  - Multiple tasks can be conducted in parallel
- Robustness:
  - The loss of one robot is not a problem
  - Each robot has a simple design
- Cheaper robots
Example 1

Robots pulling a child
Example 2

Robots getting a book
Issues with Swarms of Robots

**Swarm of robots**

100+ robots in cooperation

**Hardware design**

- Simple robots but still able to perform a task
  - 100 robots able to push won’t do much
- Reduce the cost of each unit
- Ease of manipulation
  - Updating manually the program on 100 robots is problematic

**Software design**

- The global behavior of the swarm depends of local interaction
  - Difficult to design local behaviors
- The design method has to be robust
  - Variation in number of robots
    - No central authority
  - Change of tasks during the use of robots
Problem
- Issues with Single Robots
- Proposition: Multiple Robots
- Issues with Swarms of Robots

**Biological inspiration**
- Self-organization
- Mechanisms

Swarm Intelligence

Swarm robotic
- Hand designed behaviours
- Behavioural architectures
- Learning in Swarm Robotic
Biological Inspiration

Interesting aspects

- Large number of insects
- Combination of simple actions are used to solve complex tasks
- Each insect is a simple unit (with regards to the global task)
- Any of them can die without impairing the progress of the swarm
Emergent Collective behavior

- Behavior of individual $\propto$ Behavior of neighbors
- collective phenomena $\rightarrow$ adaptive function
- not all collective movement $\rightarrow$ adaptive function
- Adaptive function
  - Simple rules
  - Local information
  - No global plan or central coordination
  - Robust (malfunction, or deviation of individuals)
The principles of self-organisation explain the biological collective phenomena where the resulting structure and functionalities greatly exceed in complexity the perceptual, physical and cognitive abilities of the participating organisms.

**Notions**

- positive and negative feedback
- equilibrium $\equiv$ attractor (dynamic systems)
Mechanisms

Cues, Signals and Stigmergy

- **cue**: unintentional (to effect others' behaviour)
  - e.g. trail in the snow
  - positive feedback: decision to follow
  - negative feedback: decision to avoid
  - Stigmergy: modification of the environment

- **signal**: intentional (to effect others' behaviour)
  - e.g. alarm cry of birds (predator)
Behaviors - Flocking and Schooling

- coordinated navigation
- tens to thousands
- local information
- no group leader
- aggregation: single superorganism
- rapid change in direction
- disperse, reunite
Behaviors - Herding and V-formation

- leader
- restrictive rules for relative motion
Foraging

- stigmergy
- path: more ants $\rightarrow$ more pheromones $\rightarrow$ more ants
- long path chosen early $\rightarrow$ wrong path
- Solution: fast evaporation
Division of labor

- specialised (soft):
  - can perform number of tasks
  - perform activity most needed
- specialised (hard):
  - perform one or few activities
- foraging, nest defence (ants)
- nektar and pollen collection (bees)
- response threshold model:
  - task stimulus $\propto$ individual’s task threshold
- dynamic task allocation
Outline

- Problem
  - Issues with Single Robots
  - Proposition: Multiple Robots
  - Issues with Swarms of Robots
- Biological inspiration
  - Self-organization
  - Mechanisms
- **Swarm Intelligence**
- Swarm robotic
  - Hand designed behaviours
  - Behavioural architectures
  - Learning in Swarm Robotic
Swarm Intelligence

Definition
Swarm intelligence is the emergent collective intelligence of groups of simple individuals i.e. collectively solve problems that are too complex for a single agent or can exhibit the robustness and adaptivity to environmental variation seen in biological agents.

Key principles
- Simple individuals (computational or physical)
- Solve complex problems
- Robust despite loss of individuals or failure
- Individuals have:
  - Sensory information; little/no memory;
  - NO global information
Optimization techniques

Particle Swarm Optimization
- Inspired from flocks of birds looking for food
- Sense neighborhood: calculate best (gbest)
- Memory: location with highest food concentration (pbest)
- Strategies: brave, conservative, swarm
- Optimization: space of food concentrations, solutions are birds

Ant Colony Optimization
- Stigmery
- Artificial ants walk on edges
- Choose an edge
- Depose pheromones on the way back
Example - Routing algorithm

Problem
- Large network composed of heterogeneous elements
- Want to obtain
  - High bandwidth
  - Low latency

Interest of swarm intelligence?
- Scalability
- Fault tolerance
- Adaptation
- Modularity
- Autonomy

Ant-based
- Ants travel in the network back and forth
- During trips, they compute the time needed
- The routing tables are updated thanks to these information
Challenges of Swarm Intelligence

Reverse engineering
- Find individual behaviour rules
- Simple rules
- Hand-designed or evolved

Stable emergent behaviour
- Dynamic systems theory
- Negative and positive interactions
- Move in space and time
Outline

Problem
- Issues with Single Robots
- Proposition: Multiple Robots
- Issues with Swarms of Robots

Biological inspiration
- Self-organization
- Mechanisms

Swarm Intelligence

Swarm robotic
- Hand designed behaviours
- Behavioural architectures
- Learning in Swarm Robotic
## Hand designed behavior

### Principle
- Find minimal interactions necessary to realize global behaviors
- Write the behaviors as simply as possible

### Minimal method
- Iterate design and test (black magic)
- Possibly long process
- Verify that it is working on various conditions
  - Number of robots
  - Noise in environment
Example

- **Didabots**
  - Physical constraint: blind in front

- **Simple behavior**
  - By default move forward
  - If obstacle, backward and then turn on the right
A library of behaviours
- ex.: obstacle-avoider, target-follower, go-recharge, find-target, pick-up object...

A system to select the right behaviour
- Structure et hierarchy between behaviours
Subsumption architecture (1/2)

- Basic behaviour
- Abstraction level
Methodology

1. Build Minimal System
2. Test System on Robot
3. Evaluate Results
4. Add New Behavioral Competence
5. Done
Subsomption architecture (2/2)
## The robot
- 2 wheels
- 4 IR sensors in the front
- 2 bumper sensors in the front
- 2 light sensors

## The task
- Go towards the light while avoiding obstacles
- Which behaviours?
- In which order?
Exercise (2/2)
Example
Back to roomba
"The Nerd Herd"

- Combination of 4 or 5 basic behaviours
  - **Homing**: navigation to a specific place
  - **Dispersion**: maintain distance to avoid conflicts
  - **Aggregation**: get together to make a group
  - **Self-wandering**: avoid collisions
  - **Following**: capacity to follow another robot (optional)

- Résultats: 20 robots do various tasks in a dynamic environment (flocking et foraging)
Details of behaviours (1/2)

Aggregate:
- Whenever nearest agent is outside $d_{aggregate}$
  . Turn toward the local centroid$_{aggregate}$, go
- Otherwise, stop

Dispersion:
- Whenever one or more agents are within $d_{disperse}$
  . Move away from Centroid$_{disperse}$

Follow:
- Whenever an agent is within $d_{follow}$
  . If an agent is on the right only, turn right
  . If an agent is on the left only, turn left

Home:
- Whenever at home
  . Stop
- Otherwise, turn toward home, go
Avoid-Kin:
  - Whenever an agent is within $d_{\text{avoid}}$
    - If the nearest agent is on the left
      - Turn right
    - Otherwise, turn left

Avoid-everything-else:
  - Whenever an obstacle is within $d_{\text{avoid}}$
    - If obstacle is on right only, turn left
    - If obstacle is on left only, turn right
    - After 3 consecutive identical turns, backup and turn
    - If an obstacle is on both sides, stop and wait
    - If an obstacle persists on both sides, turn randomly and back up

Move-around
  - Otherwise move forward by $d_{\text{forward}}$, turn randomly
Comments

Team
- If a team member fails no agents will take its place
- Coordination: not planned

Missions
- Mission changes: complete new set of behaviour is programmed
- Multiple missions: not planned
Motivation
- Behavior robust to robot failure and noise
- Change of team composition
- Dynamic environments

Class of problems
- Heterogeneous team of robots are in a task oriented mission
- Robots are programmed to perform a sub-set of the tasks
- Select proper actions so that:
  - High level fault tolerance
  - Adaptivity
  - Efficiency
L-ALLIANCE - 1: Principles

**Behavior based**
- High level of fault tolerance
- Adaptivity

**Coordination by multiple agents**
- Prediction of the final result is hard
- Optimal scheduling methods are hardly applicable on real robots
- Relies on:
  - Set of pre-programmed behaviors
  - Motivation elements:
    - Impatient
    - Acquiescence
  - Learn estimate of own performance
  - Learn estimate of other robots performance
L-ALLIANCE - 2: The algorithm

- Active learning
  - Training
  - max patient
  - min acquiescent

- Adaptive learning
  - Perform mission
  - Parameters update
  - Task ordering
## Tasks

- Robots $R = \{r_1, r_2, ..., r_n\}$
- Tasks $T = \{task_1, task_2, ..., task_m\}$
- $b$ behaviours set per robot $A = \{a_{i1}, a_{i2}, ...a_{ib}\}$
- $H : A_i \rightarrow T$
- $H = \{h_1(a_{1k}), h_2(a_{2k}), ..., h_n(a_{nk})\}$

## The parameters

- $\delta_{fasti,j}(t)$:
  - Rate of impatience when no other robots perform $task_j$
  - Task ordering
- $\delta_{slowi,j}(k, t)$:
  - Rate of impatience when $robot_k$ performs $task_j$
  - Dynamic task reallocation
- $\psi_{i,j}(t)$:
  - Time of acquiescence (maintain activity $a_j$)
  - Dynamic task reallocation
Control strategies

I Distrust Performance Knowledge about Teammates
Robot uses only its own knowledge about the task
Becomes impatient if other robot is slower than itself

II Let the Best Robot Win
Let other faster robots perform the task
Becomes impatient if other robot is slower than the internal estimate

III Give Robots a Fighting Chance
Use other robots estimate of completion time
Compete only if better than others estimate

Task ordering strategies

Longest Task First

Modified Shortest Task First
Selection in the action a robot can do better than the other robots

Modified Random Task Selection
Baseline
L-ALLIANCE - 6 : Results

- Should do first the longest where robot is the best
- Then start by the shortest
<table>
<thead>
<tr>
<th>Application domain</th>
<th>Metric description</th>
</tr>
</thead>
<tbody>
<tr>
<td>‘Mock’ hazardous waste clean up</td>
<td><strong>Time of task completion</strong> <strong>Total energy used</strong></td>
</tr>
<tr>
<td>Box pushing</td>
<td><strong>Perpendicular dist. pushed by unit time</strong></td>
</tr>
<tr>
<td>Janitorial Service</td>
<td><strong>Time of task completion</strong> <strong>Total energy used</strong></td>
</tr>
<tr>
<td>Bounding overwatch</td>
<td><strong>Distance moved per unit time</strong></td>
</tr>
<tr>
<td>Formation-keeping</td>
<td><strong>Cumulative formation error</strong></td>
</tr>
<tr>
<td>Simple multi-robot manipulation</td>
<td><strong>Number of objects moved per unit time</strong></td>
</tr>
<tr>
<td>Cooperative tracking</td>
<td><strong>Avg. numbers of targets observed</strong></td>
</tr>
<tr>
<td>Multi-vehicle production dozing</td>
<td><strong>Quantity of earth moved per unit time</strong></td>
</tr>
</tbody>
</table>
Learning

Limitations of hand design approach
- has to be able to somewhat predict the useful behaviors
- has to design the basic behaviors
- => addressing very different environment is not possible

Proposition
- Multi robots in unknown environment
- Only the task to be realized is specified
- => Autonomous design method
Reinforcement learning

- Set of states and actions
- Rewards attributed to specific states
- Learn the policy that maximize the rewards
- \( \Rightarrow \) Real robots operate in continuous and noisy environments
  - But methods have been proposed to address this point

Evolutionary robotic

- Focus for this course
- Use of evolutionary algorithms in robots
- Adaptable to a large range of tasks
- Works on real robots