Strategies for Maintaining Large Robot Communities

S. English, J. Gough, A. Johnson, R. Spanton,
J. Sun, R. Crowder, and K.-P. Zauner

School of Electronics and Computer Science
University of Southampton, U.K.

Motivation

Many organisms solve tasks collaboratively: social insects, flocks of birds, schools of fish. . .
/Potential applications for robot swarms include/
Short term:

- Marking the perimeter of a chemical spill
- Marking an accident on a high-way
  Long term:
- Environmental monitoring
- Assembly of structures

Obstacles to Research:

- Cost of acquiring a swarm of robots
- Practical difficulty of dealing with large numbers of robots

Outline

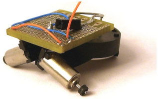
1. Hardware

2. Software
   - Task Allocation

Implementation

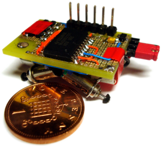
Scalable design:

- Off-the-shelf components
- Low component count
- Simple assembly ⟵ dominating cost factor

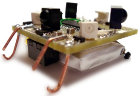


Design solution:

Motors used to vibrate mobile phones in direct drive



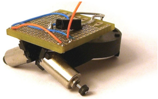
- Cheap (mass produced)
- Prepared for surface mounting
- No need for gears



Implementation

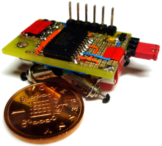
Scalable design:

- Off-the-shelf components
- Low component count
- Simple assembly ⟵ dominating cost factor

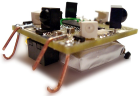


Design solution:

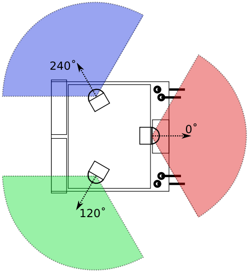
Motors used to vibrate mobile phones in direct drive



- Cheap (mass produced)
- Prepared for surface mounting
- No need for gears



Communication

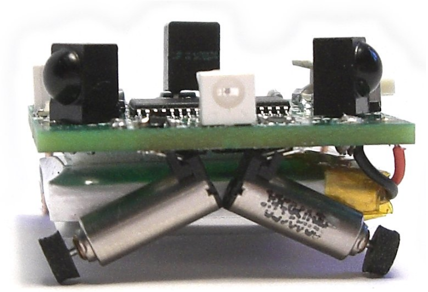


- Infrared Communication
- Broadcasting only
- Speed 588 B/s
- Range in motion ≈10 cm
- Multiple frequency-shift keying
  - 18 Frequencies / 4 bit chips
  - Passive filtering of IR noise from motion

Developed with an MSP430F2001 with comparator, but implemented with an op-amp version to reduce cost.

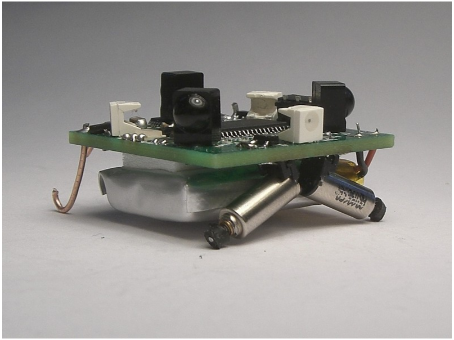
Current Version

- Size: 30×28×12 mm
- Processor: MSP430F2254—16 MHz
- Memory:
  - 512 B RAM—256 B free
  - 16 kB Flash—10 kB free
- Battery: 320 mAh
  - ≈ 1.5 h activity
- IR Ground Sensor
- On-board charging circuit



- IR receivers can be used to directionally detect ambient light
- Reflection of IR broadcast can be used to detect obstacles

Cost per robot for 1000

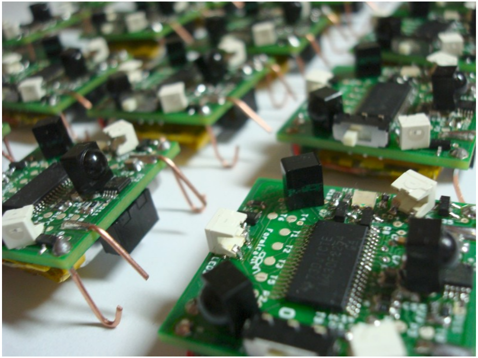


Processor	£1.73
Battery	£0.75
Battery charger	£0.46
2× Motor	£0.50
Other components	£3.12
PCB and assembly	£8.19
Total	£14.75

Time for fabrication: 6 weeks.

Summary

Large robot swarms are "just around the corner"!

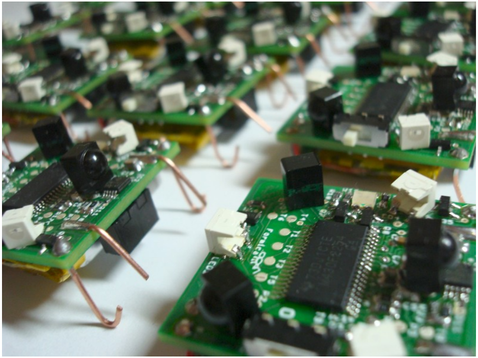


But do we really want them?

- Tending one robot can take up a lot of time.
- How about 500?

Summary

Large robot swarms are "just around the corner"!



But do we really want them?

- Tending one robot can take up a lot of time.
- How about 500?

Outline

1. Hardware
2. Software
   - Task Allocation

General Requirements

- Realistic
- Scalable
- Robust
- Responsive
- Controllable

General Requirements

- Realistic
  - Can run on simple hardware
  - Can cope with unreliable communication
- Scalable
  - Near constant resource requirement on each robot
  - User interacts only with a small number of robots
- Robust
  - Should not loose tasks from fluctuations
  - Self-regulate task allocation if population size changes
- Responsive
  - Quick distribution of tasks in a blank swarm
  - Quick return to desired task distribution after disturbance
- Controllable
  - State of swarm can be changed by interacting with a few robots

What is realistic?

Constraints:

- Memory (especially RAM)
- Communication speed
  - No global knowledge!
- Communication reliability
- Processing speed
- Limitations of tools
  - Code not relocatable

Possibilities:

- Counters
- Timers
- Unique ID numbers

Limitations:

- Security not considered

Plasmids ⟶ . . . point to a possible solution.

Bacteria face a similar problem: they are under pressure to keep their DNA short for fast replication

Plasmids

- Carry genes for additional functionality:
  - resistance to antibiotics
  - defensive toxins
  - metabolism of nutrients
- Readily transferred from one bacterium to another
- Whole genomes can be transferred

Robots could mimic this:

- New robots have only the firmware
- A few robots receive additional software for task-specific functionality
- Most robots acquire task-specific software from other robots in the swarm

Plasmids ⟶ . . . point to a possible solution.

Bacteria face a similar problem: they are under pressure to keep their DNA short for fast replication

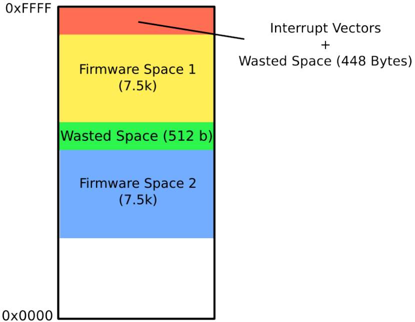
Plasmids

- Carry genes for additional functionality:
  - resistance to antibiotics
  - defensive toxins
  - metabolism of nutrients
- Readily transferred from one bacterium to another
- Whole genomes can be transferred

Robots could mimic this:

- New robots have only the firmware
- A few robots receive additional software for task-specific functionality
- Most robots acquire task-specific software from other robots in the swarm

Scalable Firmware Distribution



- After buildup of image, overwrites IVT and jumps to reset vector
- Odd and even releases need to be linked for higher and lower image, respectively

Supports self-flashing with new firmware from IR download.

- Robots regularly transmit firmware packages with
  - firmware version
  - 16B block of the firmware image
  - block-address
- Robots can request packages in a broadcast message
- Robots with a suitable firmware version add requests to a short (3) ring buffer and broadcast accordingly
- Transmission time 7–10 min

Task Allocation

Problem

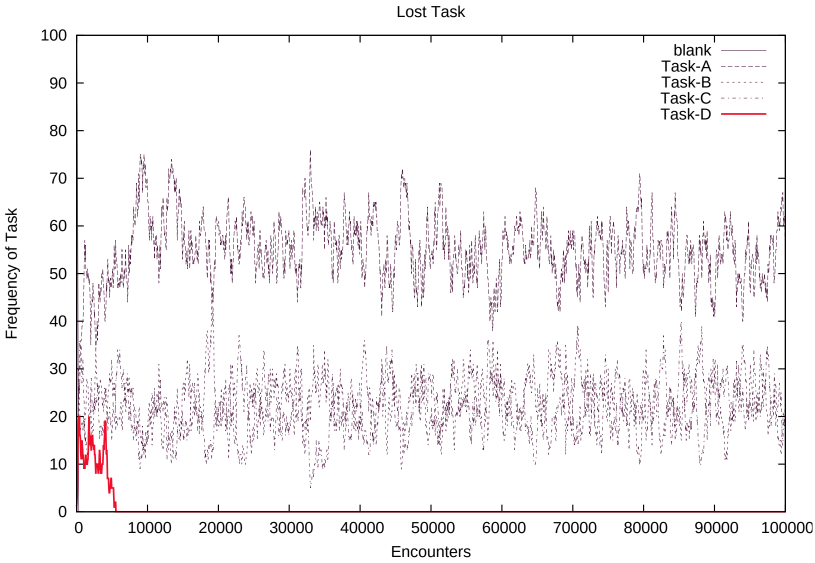
- Allocate given proportions of a swarm to different tasks

Assumptions

- Any robot can only carry one task.
  $\Rightarrow$ Task switching requires presence of other robot with new task
- Communication by short-range broadcast only
- Fast random motion of robots (well mixed)
- Only one robot per task is pre-programmed

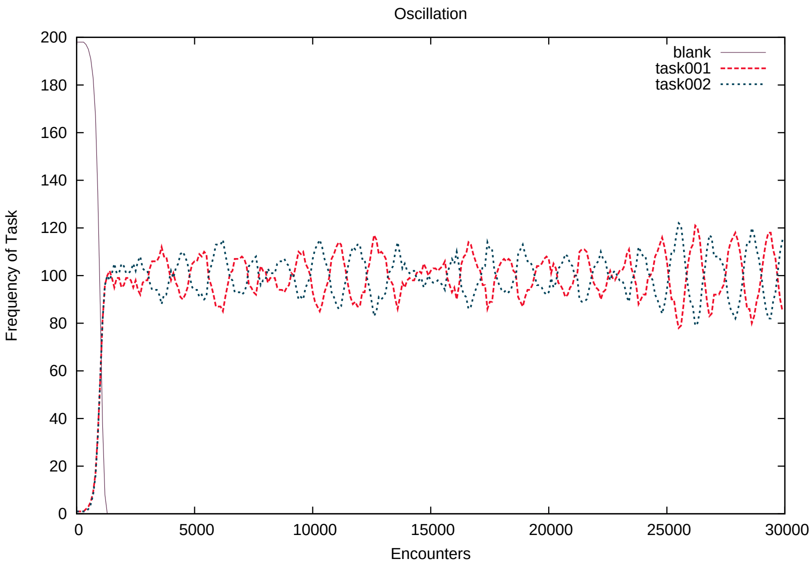
These assumptions are more restrictive than what the hardware can support.

A solution should avoid. . .



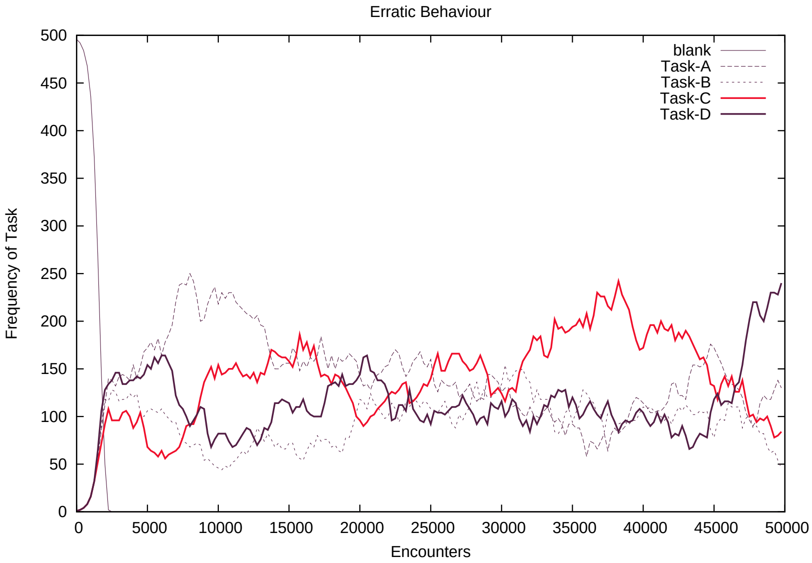
Tasks lost by fluctuation

A solution should avoid. . .



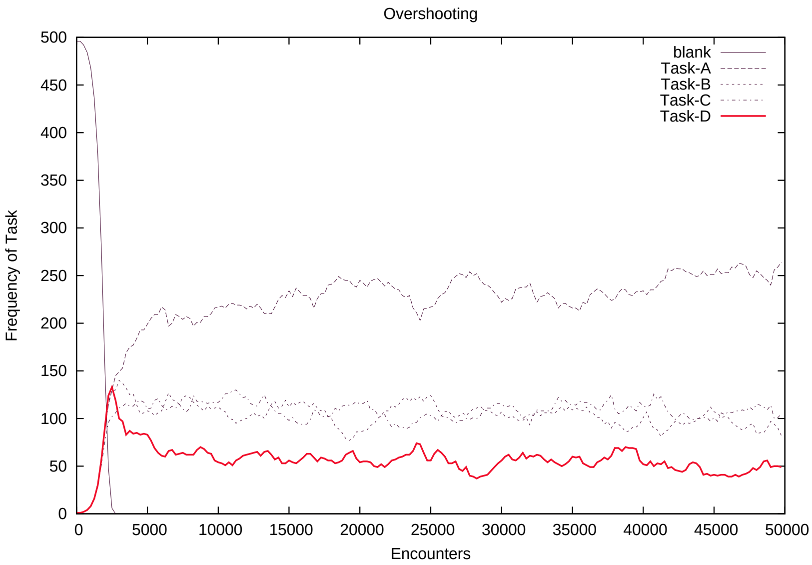
Oscillations (unnecessary overhead for changing tasks)

A solution should avoid. . .



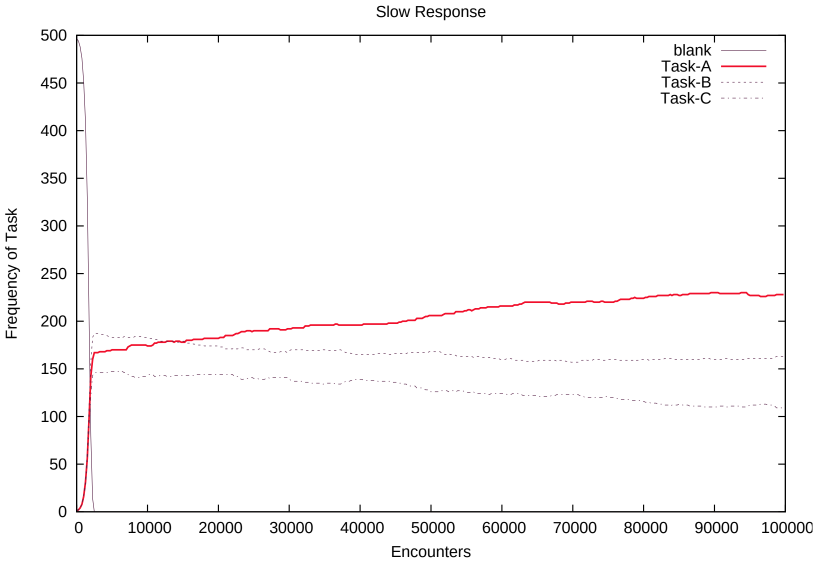
Erratic behaviour

A solution should avoid...



Overshooting

A solution should avoid. . .



Slow response

Proposed Method

Principle

- Each task is associated with a set of labels
- Robots randomly choose one of the labels associated with their task as the active one and broadcast it; they frequently change the label they are using
- Interactions among the robots equillibrate frequency of active labels in the swarm
- The number of labels associated with a task will determine its proportion in the swarm

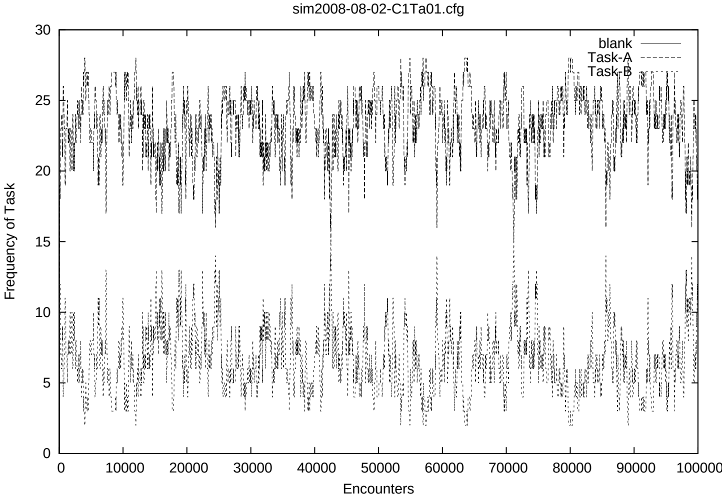
Interaction of Robots

- If a robot receives a broadcast with the same label it currently broadcasts it goes into a receptive state
- A receptive robot will take on the next task it encounters and turn non-receptive.

Simulation Results

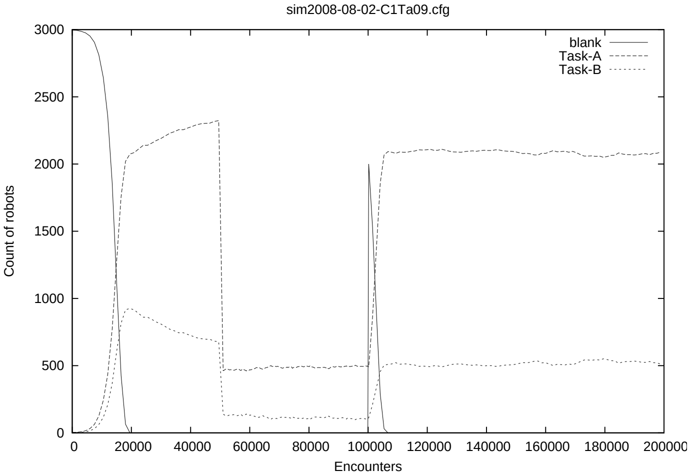
- Simulations start with a pool of blank robots
- Each task is seeded by a single robot at $t = 0$
- Progress is plotted against pair-wise encounters (x-axis)
- Simulation runs take only a few seconds
- For clarity a typical result from a single run is shown
- Simulations carried out with swarm sizes from 20 to 3000 robots

Simulation Results: small scale



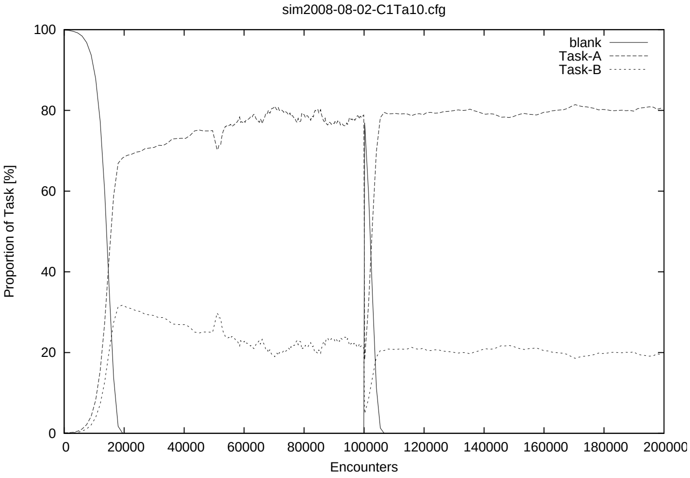
- Population 30
- Desired allocation A: 80%, B: 20%

Simulation Results: Robustness I



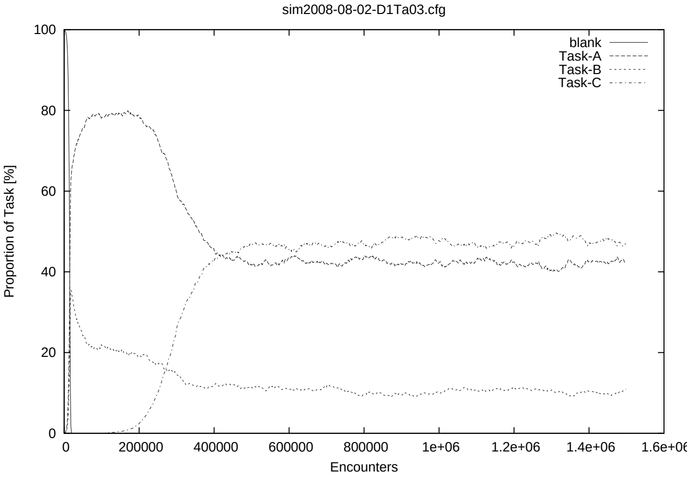
- Start population 3000
- $t = 50000 \longrightarrow 80\%$ of the swarm is destroyed
- $t = 100000 \longrightarrow 2000$ blank robots are added

Simulation Results: Robustness II



- Start population 3000
- $t = 50000 \longrightarrow 80\%$ of the swarm is destroyed
- $t = 100000 \longrightarrow 2000$ blank robots are added

Simulation Results: Control



- Population: 3000; desired initial allocation A: 80%, B: 20%
- At $t = 100000$ one robot with task C and a desired allocation of 50% is introduced

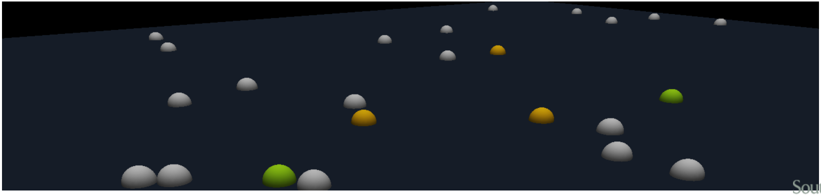
Test Migration

Problem

A number of self-test programs should migrate among the population and be present in a low copy number $< 1\%$, but not disappear.

Outline of Approach

- Robots have unique IDs
- A robot will delete its own test software if it was handed over twice successfully
- A robot will delete its own test software if it encountered several other robots with the same test



Concluding Remarks

- Swarms of hundreds or even thousands of robots are now within reach of a typical research budget
- Collaborative solutions are required to compensate for:
  - lack of hardware features
  - lack of quality control
  - lack of individual calibration
- Simulation indicates that the constraints of cheap hardware do not get in the way of scalable algorithms
- Don't trust simulations—try it on a real swarm!

See them in action...

Wednesday 17:30–19:00, Stripe Theatre Studio 2

Concluding Remarks

- Swarms of hundreds or even thousands of robots are now within reach of a typical research budget
- Collaborative solutions are required to compensate for:
  - lack of hardware features
  - lack of quality control
  - lack of individual calibration
- Simulation indicates that the constraints of cheap hardware do not get in the way of scalable algorithms
- Don’t trust simulations—try it on a real swarm!

See them in action…

Wednesday 17:30–19:00, Stripe Theatre Studio 2