University of St Andrews | FOUNDED 1413

Towards a Science of Sensor Systems Software

Simon Dobson
School of Computer Science, University of St Andrews UK

simon.dobson@st-andrews.ac.uk
http://www.simondobson.org
@simoninireland

EPSRC S4 Programme Grant (2016—2020)
http://www.dcs.gla.ac.uk/research/S4/

University of Glasgow

University of St Andrews

UNIVERSITY OF LIVERPOOL

Imperial College London

What makes sensing different

- Observing and responding to physical-world changes
  - Wireless sensor networks: temperature, pressure, humidity, proximity, target-counting, …
  - Internet of Things: the same, but with phones!! :-)
- Often building open systems
  - No traditional closed-loop control
  - Mission ~~creep~~ sprint
- Limited-capability nodes and networks
  - Lots of machine learning

From the real world

The endless stream of hardware malfunctions, programming bugs, software incompatibilities, and plain misunderstandings, combined with the time pressure of Mother Nature, has left us one year later with a meager harvest of quantitative results.

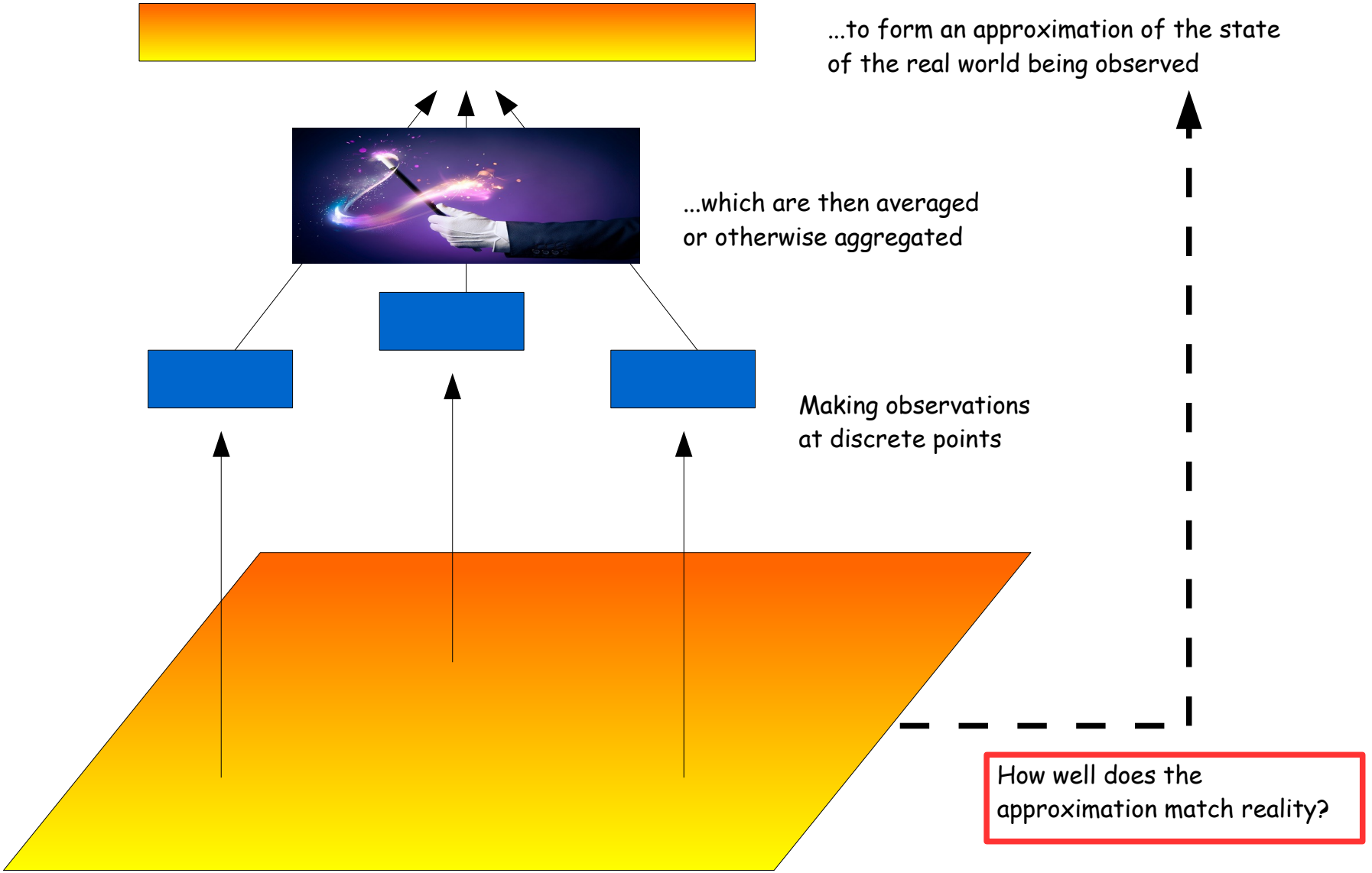
One could declare the project a failure, but on the other hand, we have learned a lot – the hard way . . . Although experiences about previous pilots have been reported, these publications in general stress technical issues like low-level network performance instead of the (basic) software-engineering problems that made running our project so difficult.

Langendoen, Baggio, and Visser. Murphy loves potatoes: Experiences from a pilot sensor deployment in precision agriculture. IEEE PDPS. 2006.

Req and spec

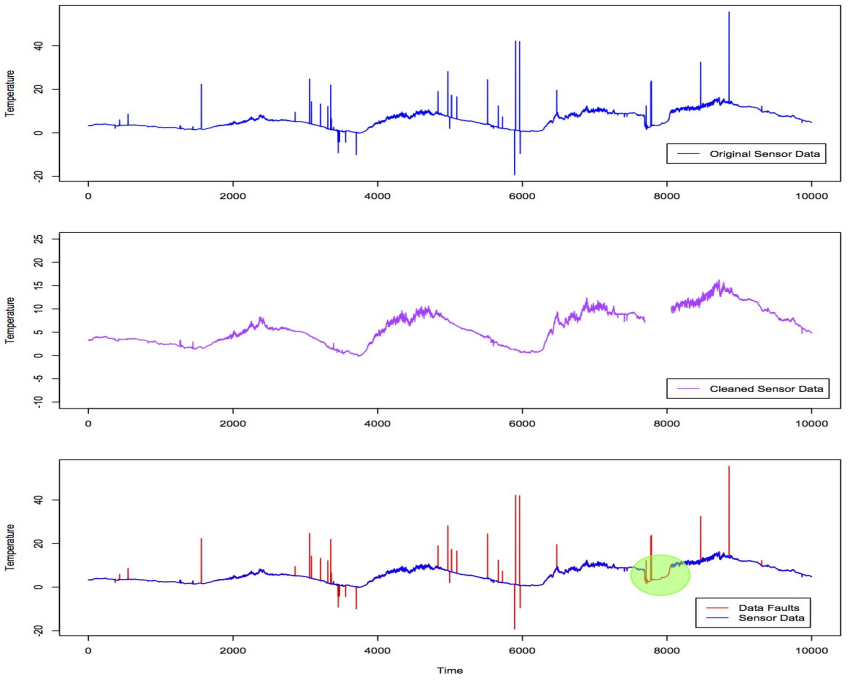
- Almost always phrased in terms of the physical environment
  - Determine temperature cline across an area
  - Observe intruders in a space
- Often not specified well/at all
  - May not even be known *a priori*
- Often encounter "best effort" deployments
  - Deploy a load of sensors with a network, see what we can find/hope we find the things we want

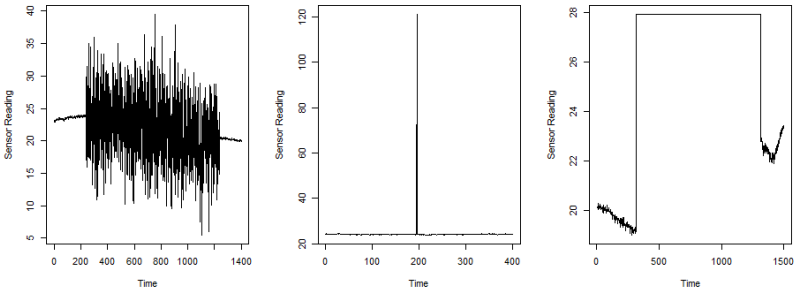
WSN design on one slide



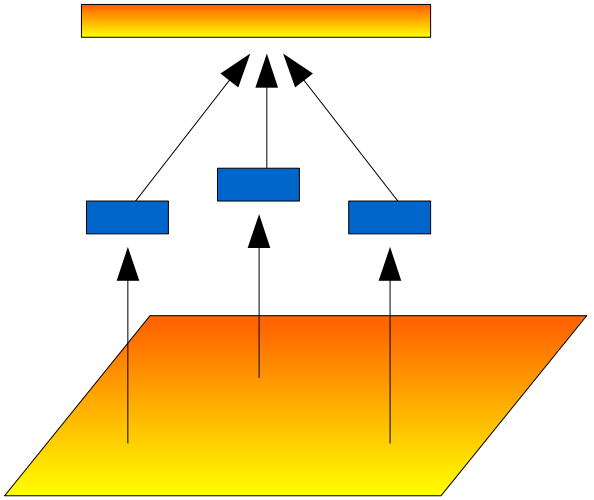
...and we often don't know

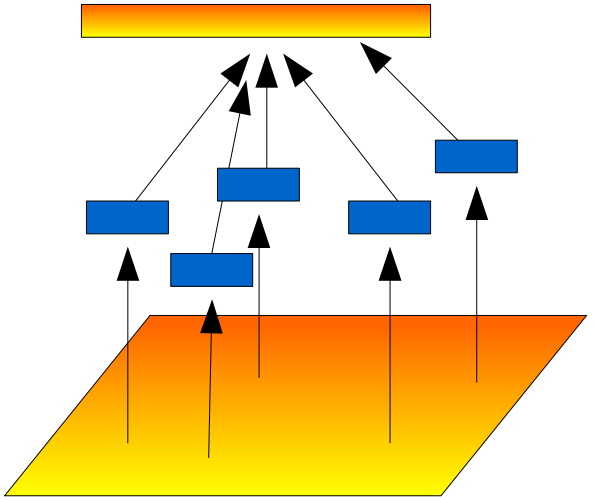
- No ground truth
  - Can't compare the *in situ* behaviour
  - Inherent noise
- Progressive degradation
  - Mechanical wear and tear
  - Partial failure
  - Malice





Basic questions – 1

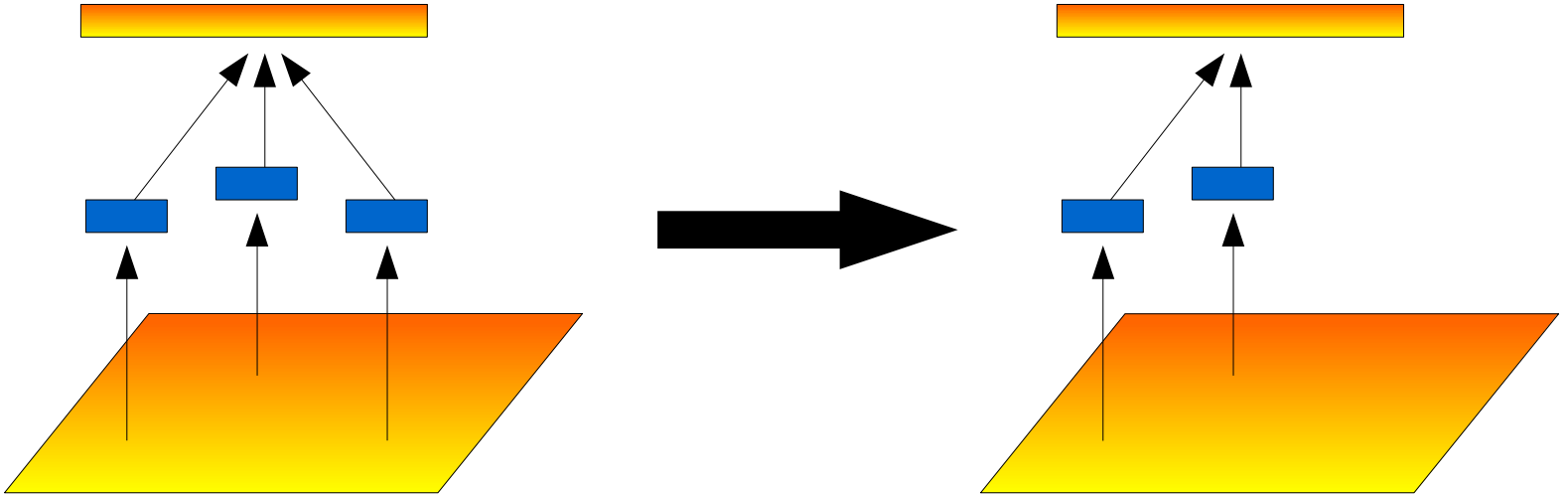




- Given two sensor layouts, which will allow more accurate conclusions?
  - Noise and overlap make this hard to answer: more is not always better

See Pianini *et alia*. Self-stabilising target counting in wireless sensor networks using Euler integration. Proc SASO 2017.

Basic questions – 2



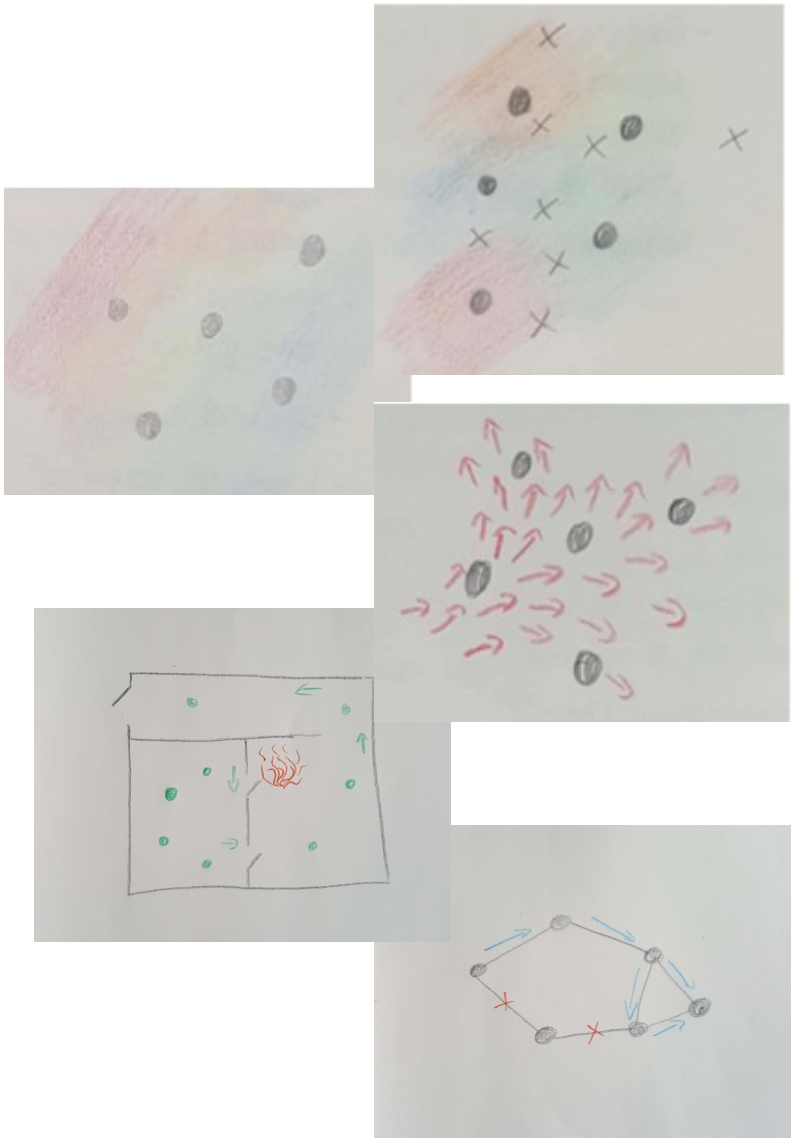
- What happens as the network degrades?
  - Long lifetimes, partial failure
  - How should confidence change?
  - How do the detectable features change?

A more engineering approach

- For a given set of interesting phenomena:
  - What is the *best* configuration to sense them?
  - How will *some specific* configuration sense them?
  - How will what we *observe* change as the network degrades or is interfered with?
  - How will our *conclusions* change?

Making a start – 1

- Defined a set of abstracted "challenge" problems
  - Realistic enough to be meaningful to solve
  - Abstract enough to be analysed/simulated
- How do different arrangements of sensors work against (known) ground truth?



Making a start – 2

- A lot of the programming approaches in WSNs
  - Extremely stylised approaches, mainly in C
  - Very poor software structuring, lots of cross-layer optimisation

And this isn't going to change soon

- Pattern-based adaptation
  - Change structure, parameters, around the core functions of sensing and actuation
  - Couple this with understanding how the low-level parameter choices affect high-level conclusions

Dearle and Dobson. Mission-oriented middleware for sensor-driven scientific systems. J. Int.Serv.App. **3**(1). 2012

Questions and abuse may now begin...