3rd Colloquium SPP OC
Quantitative Emergence

Jürgen Branke, **Urban Richter**, Hartmut Schmeck
University of Karlsruhe (TH), Institute AIFB

Moez Mnif, Christian Müller-Schloer
University of Hannover, Institute SRA

Outline

1. Motivation and project goals
2. Quantitative emergence
   - What is emergence?
   - Emergence computation
3. Generic observer/controller architecture
4. Experimental results
5. Summary and outlook

Motivation and project goals

- Increasing complexity in technical systems
- Survive attacks, breakdowns, and other unexpected events with self-x-properties

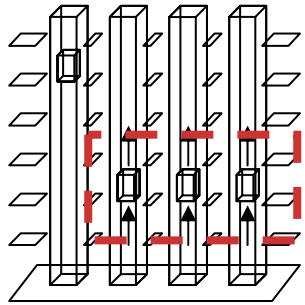
- Large number of interacting sub-systems
- Self-organisation

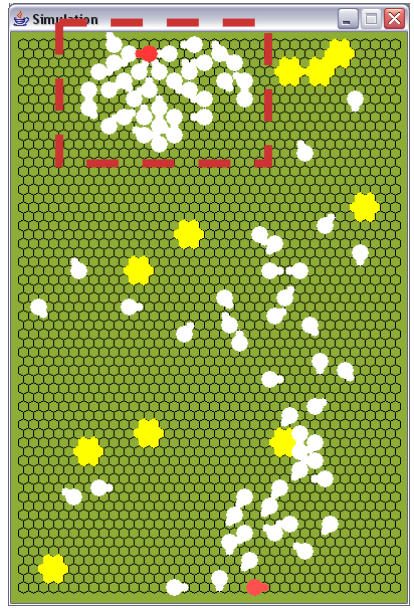
1. Development and investigation of metrics for the observation and analysis of emergence in self-organising systems (H)
2. Development and investigation of mechanisms to influence and control the effects of emergence in self-organising systems (K)
3. Realisation of tools and validation of the tools in various test scenarios

- Observer/controller architecture
- Toolbox with basic mechanisms to observe, analyse, and control emergent behaviour

What is emergence?

- "The whole is more than the sum of its parts."
- Many examples in nature, e.g. flock of birds.
- Humans decide intuitively on the occurrence of emergence.
- Precondition: large population of interacting elements without central control
- Emergence = self-organised order





- Organic computing:
  - How to build self-organising technical systems?
  - Automate the recognition of emergent behaviour (formation of order pattern).
  - Need of metrics to quantify emergence.
  - Development of mechanisms to control self-organised behaviour.

Measuring emergence by entropy

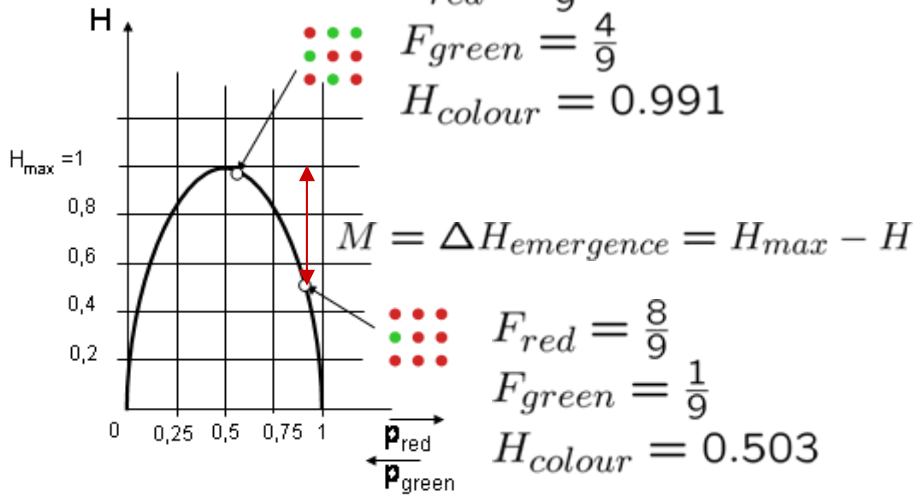
[Mnif and Müller-Schloer 2006]

- Order depends on the selection of certain attributes

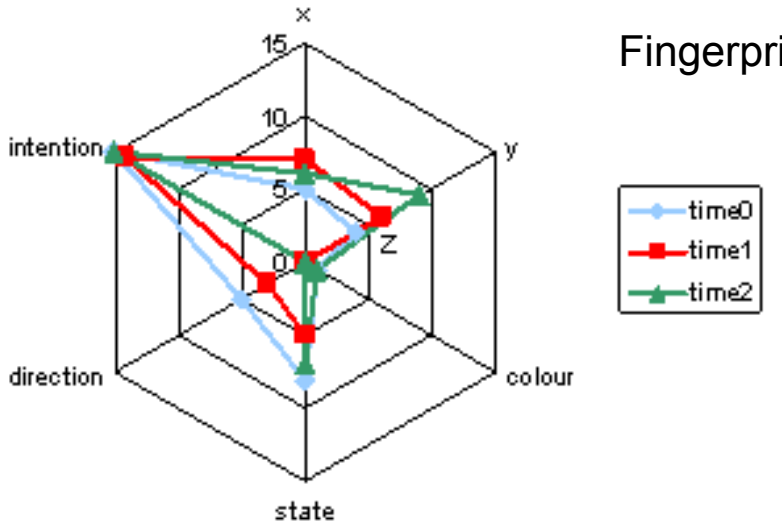
- Entropy as a well-known metric of order
  - Lower entropy ↔ higher order
  - Higher entropy ↔ lower order
- Make use of the statistical definition of Shannon's entropy

$$H_A = -\sum_j p_j \cdot \mathrm{ld} p_j$$

- Maximum entropy if the attribute values are equally distributed

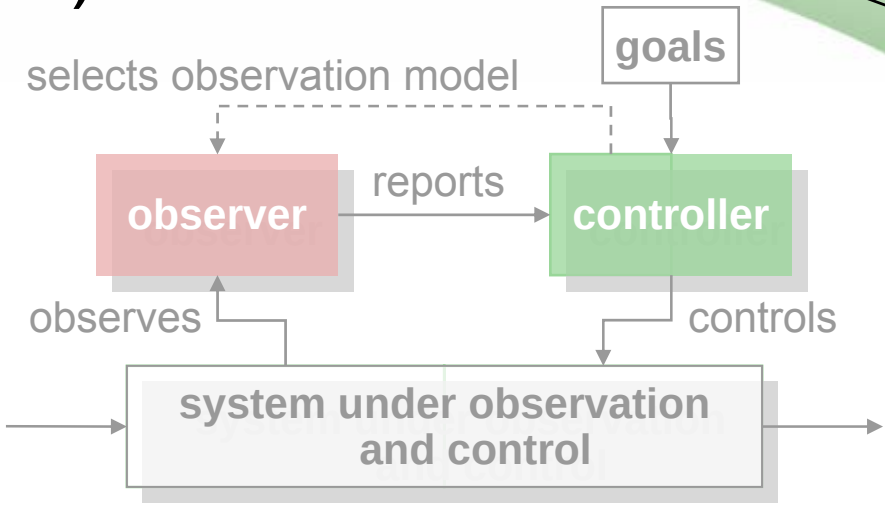


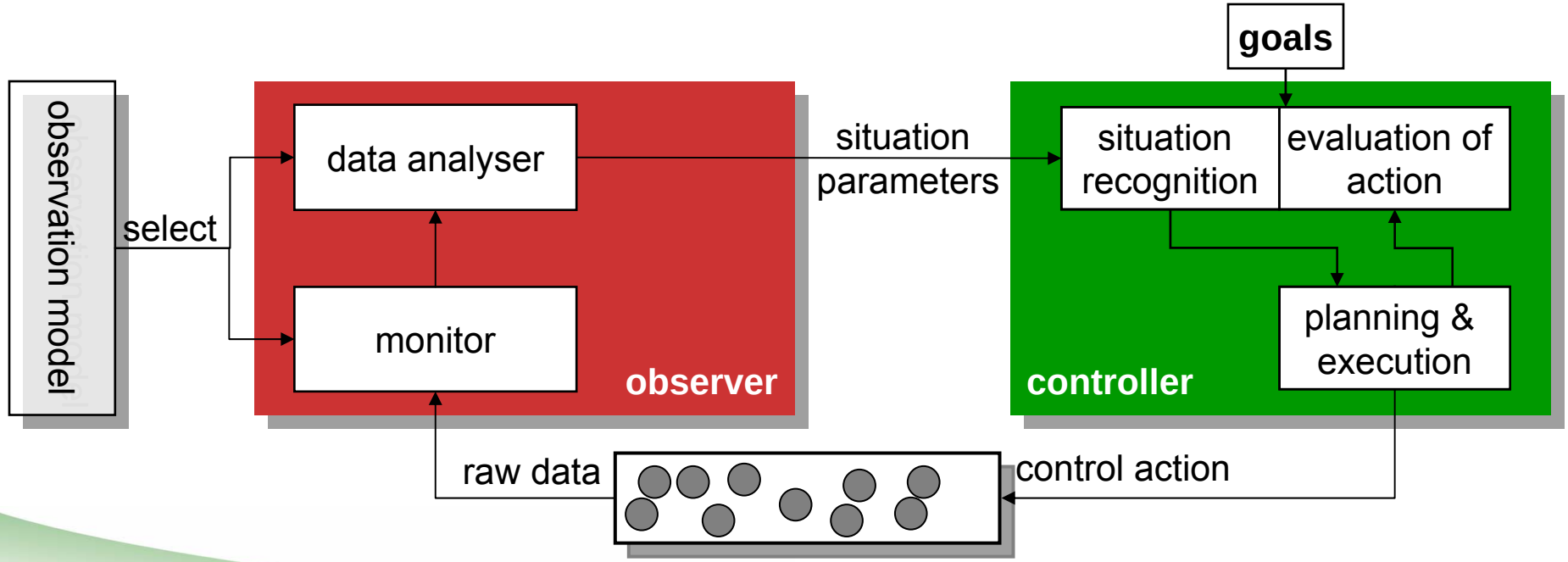
Fingerprint



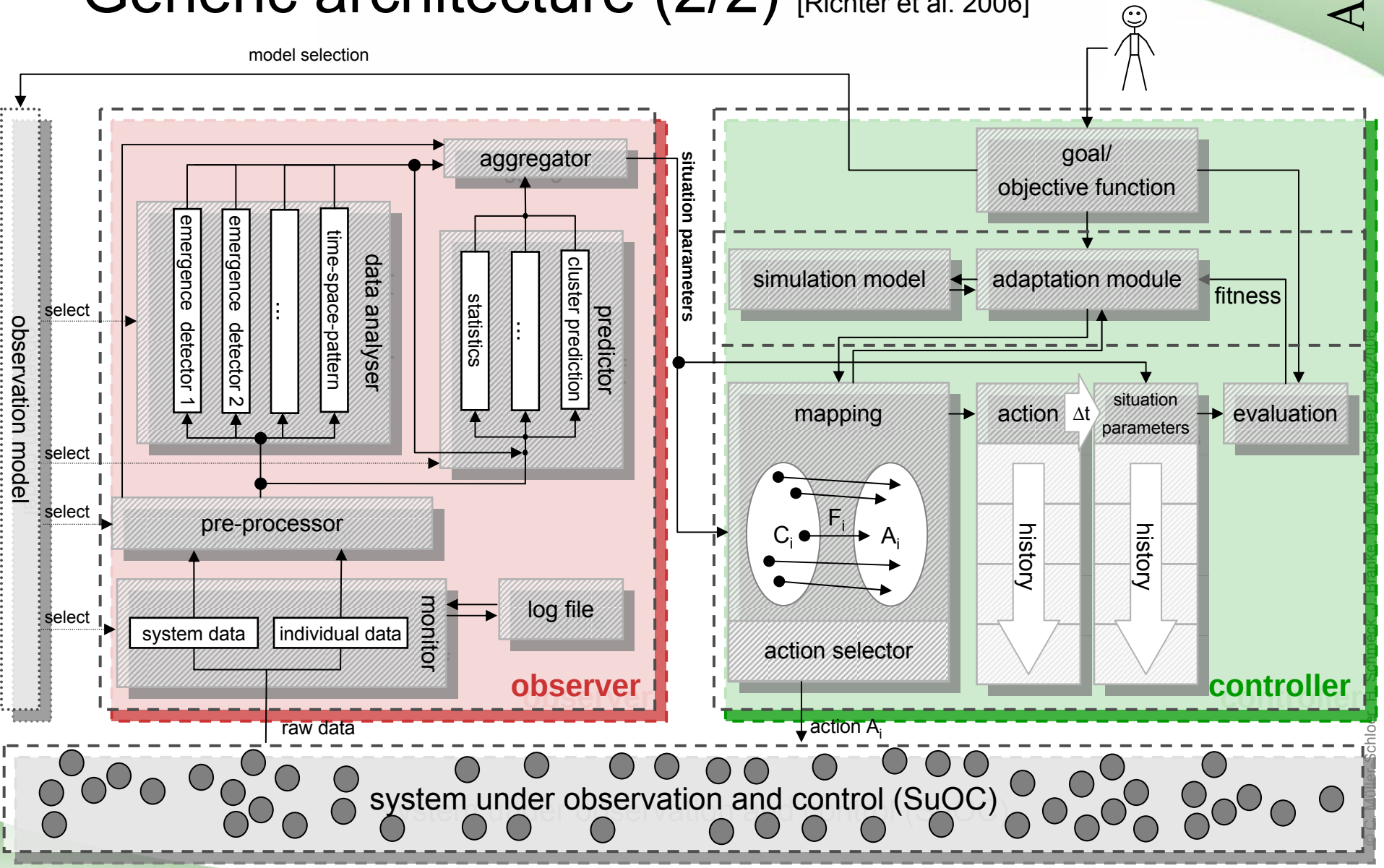
Generic architecture (1/2)

- Distributed and/or central observer/controller architecture
- Multi-level organisation



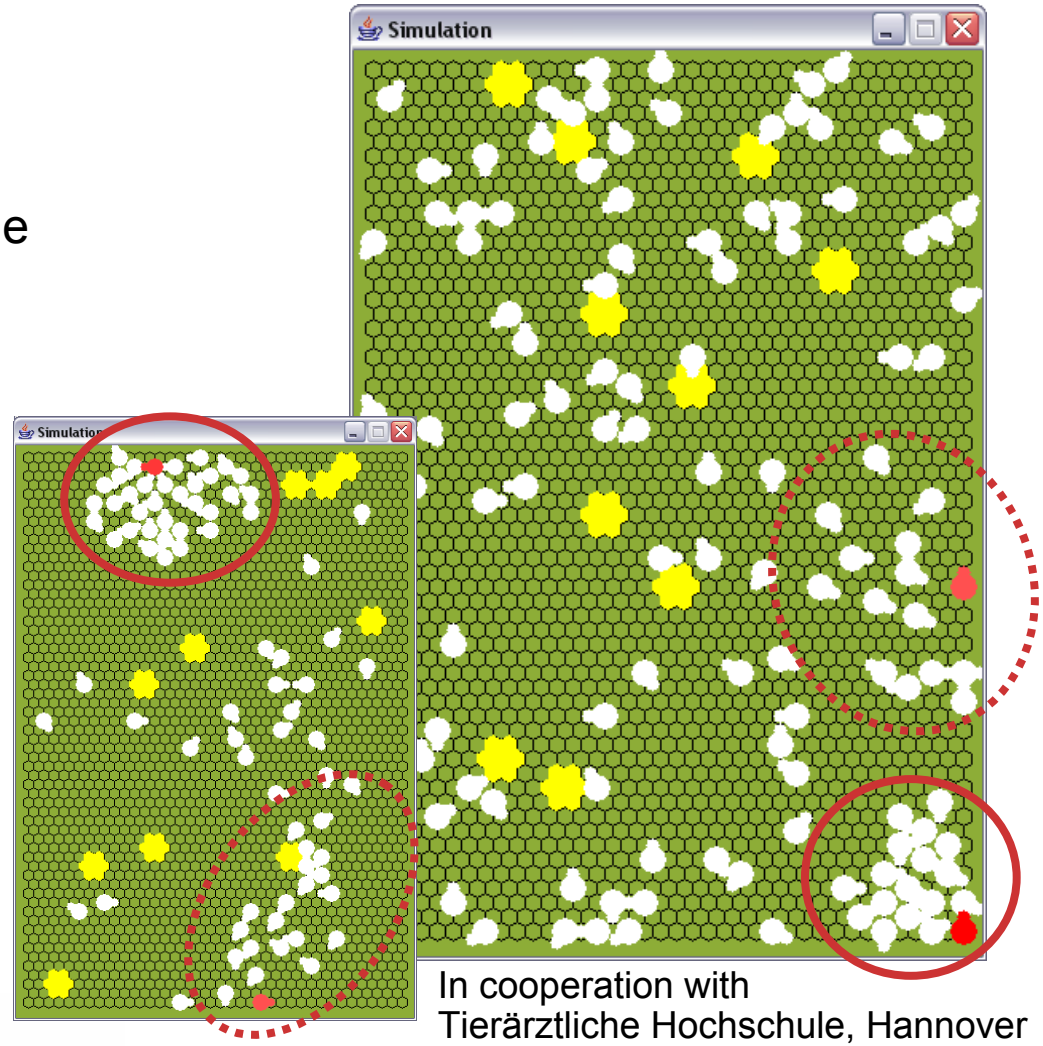


Generic architecture (2/2) [Richter et al. 2006]



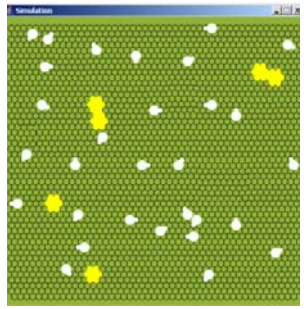
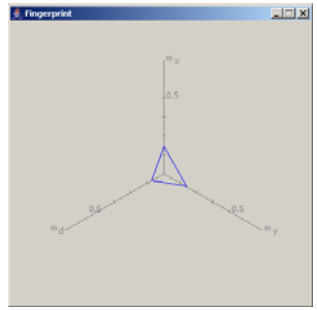
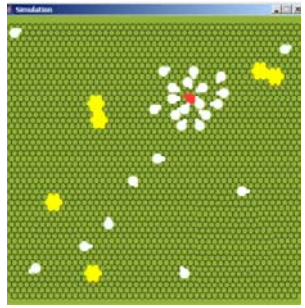
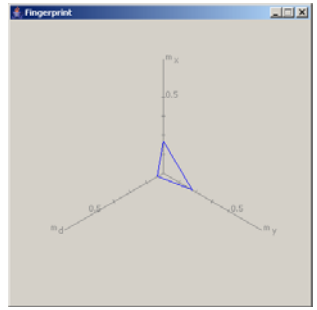
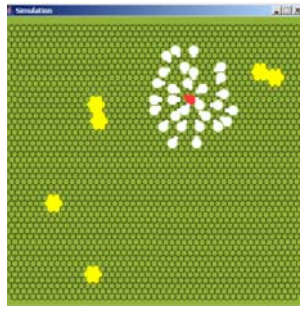
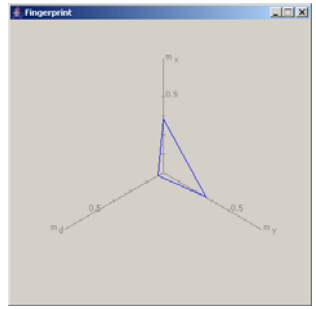
Experimental environment

- Explain/control the collective behaviour of densely packed chickens in cages
- Autonomous agents with simple rules
- Emergent behaviour is spatial (clustering)
- Observation of 3 attributes
  - X-coordinate
  - Y-coordinate
  - Direction (heading)
- Control mechanisms
  - Noise
  - Feed

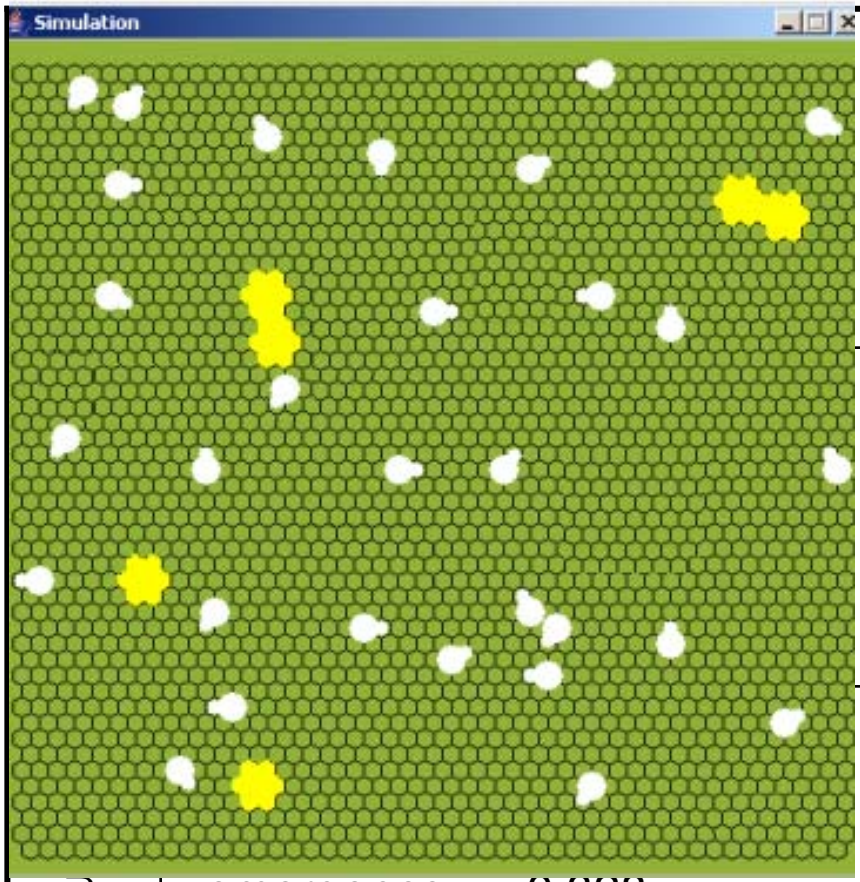


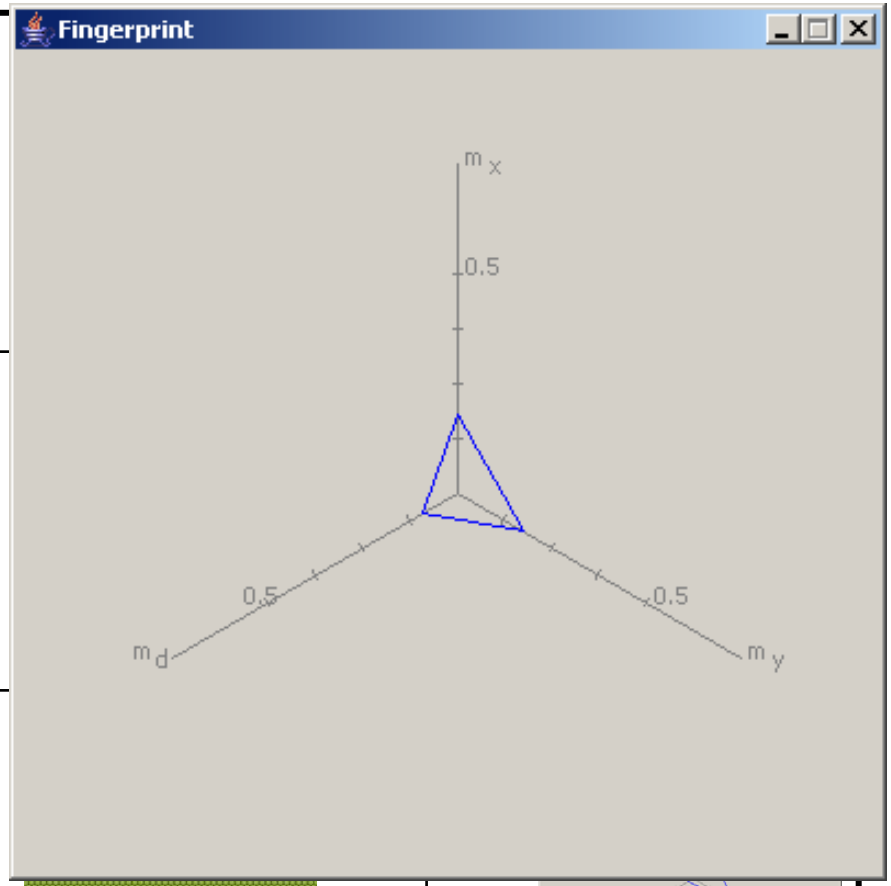
In cooperation with
Tierärztliche Hochschule, Hannover

Experimental results

Pattern 1	No cluster (no order) • emergence-x= 0.181 • emergence-y= 0.177 • emergence-d= 0.091		
Pattern 2	Small cluster + noise • emergence-x= 0.226 • emergence-y= 0.237 • emergence-d= 0.046		
Pattern 3	All chicken form a cluster • emergence-x= 0.359 • emergence-y= 0.328 • emergence-d= 0.041		

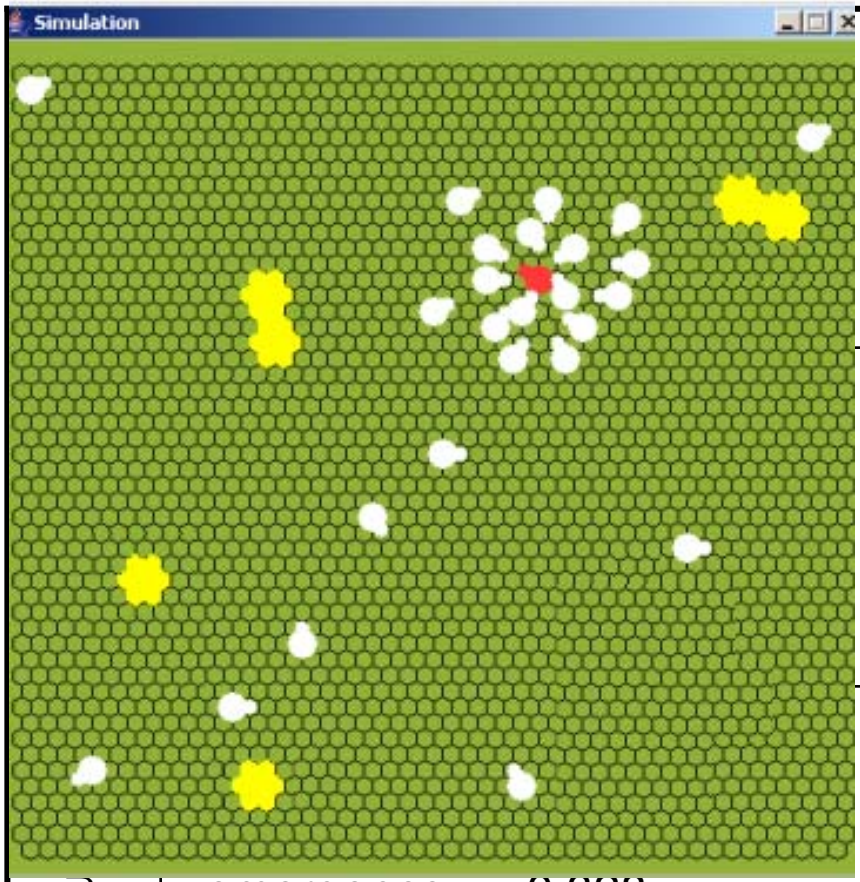
Experimental results

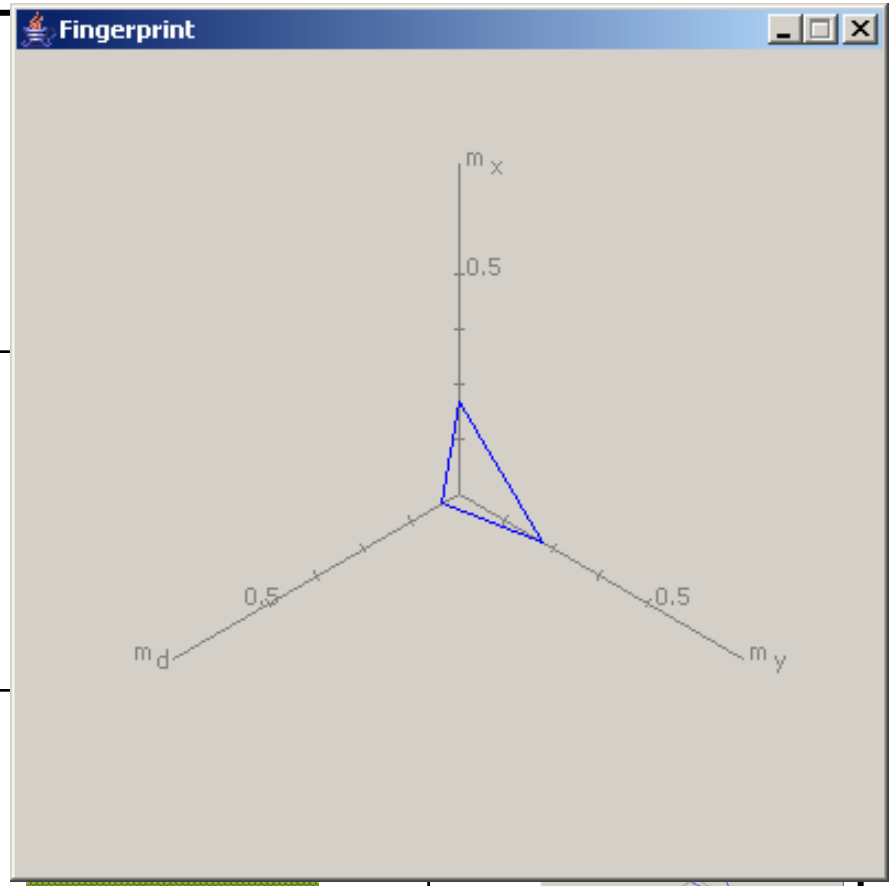




- emergence-y= 0.328
- emergence-d= 0.041

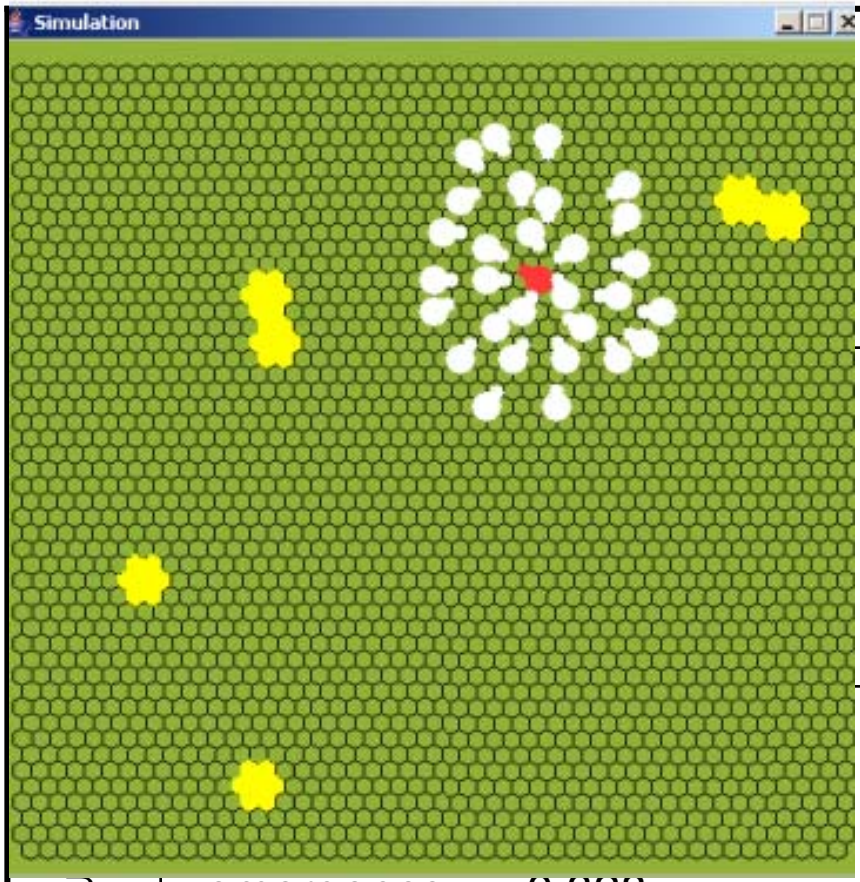
Experimental results

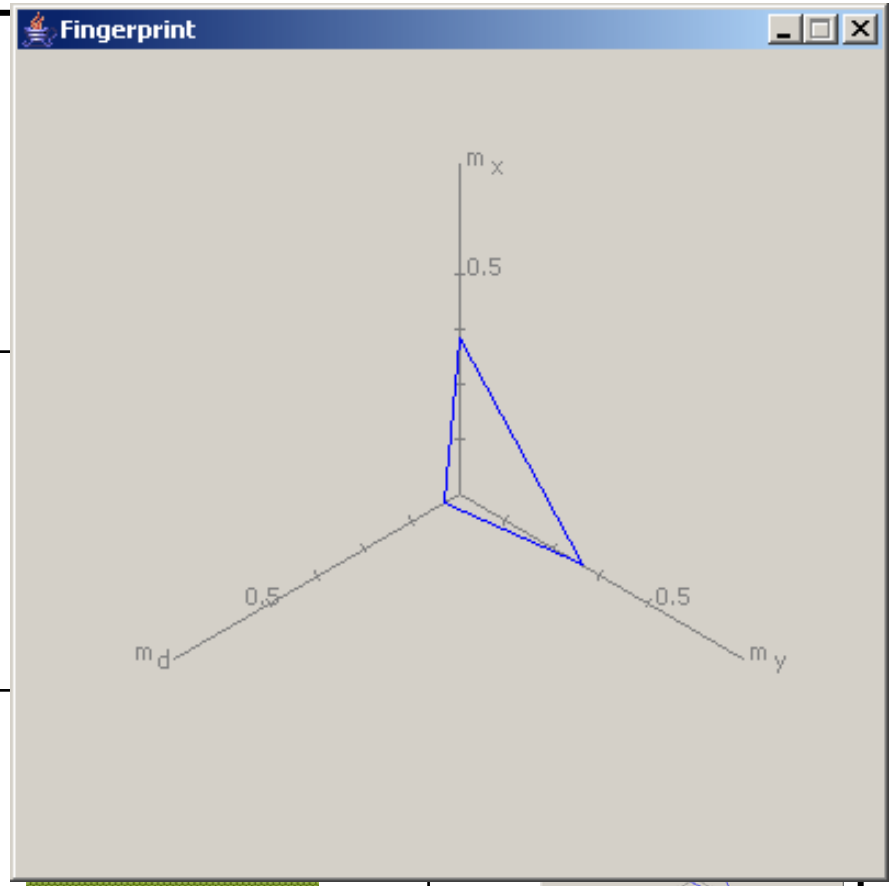




- emergence-y= 0.328
- emergence-d= 0.041

Experimental results

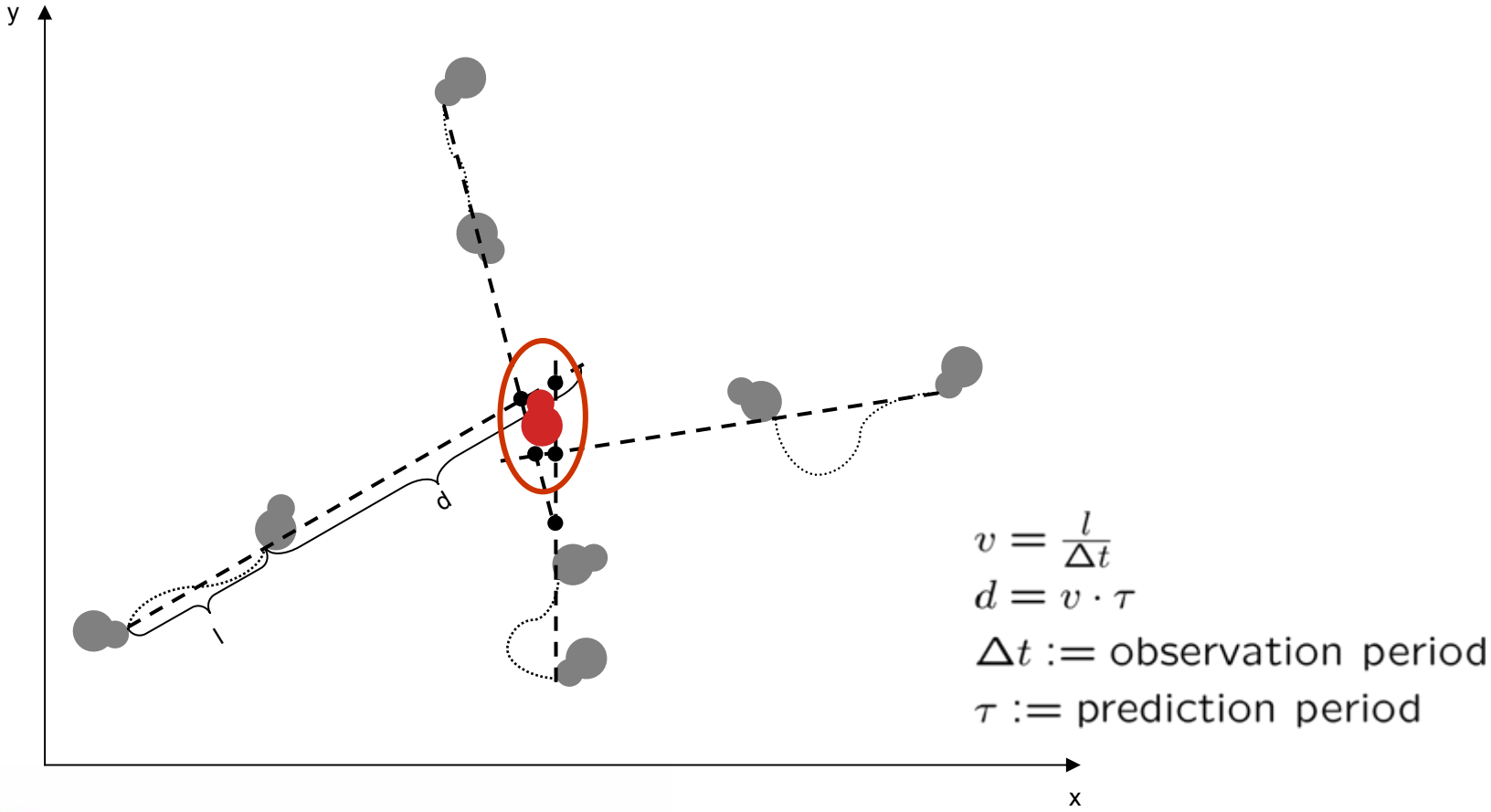




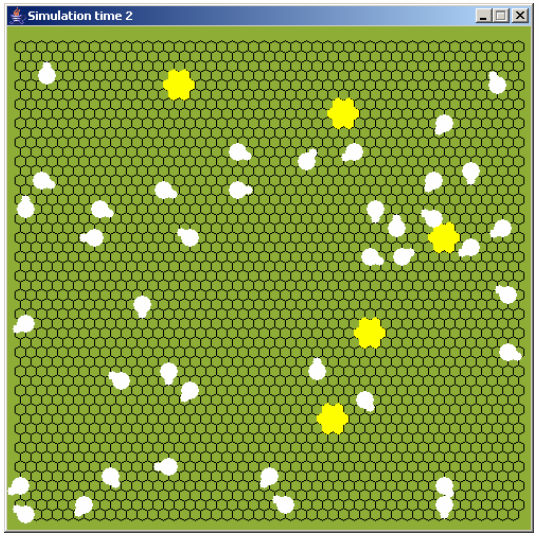
- emergence-y= 0.328
- emergence-d= 0.041

Emergence prediction

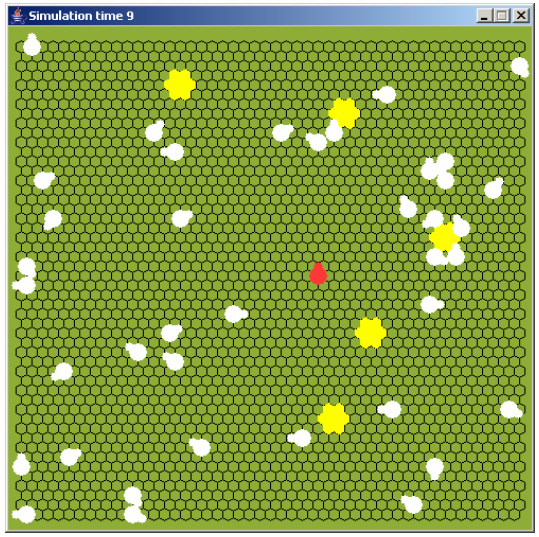
- Prediction of the chicken positions with trajectory computation
- Prevention of unwanted behaviour in time



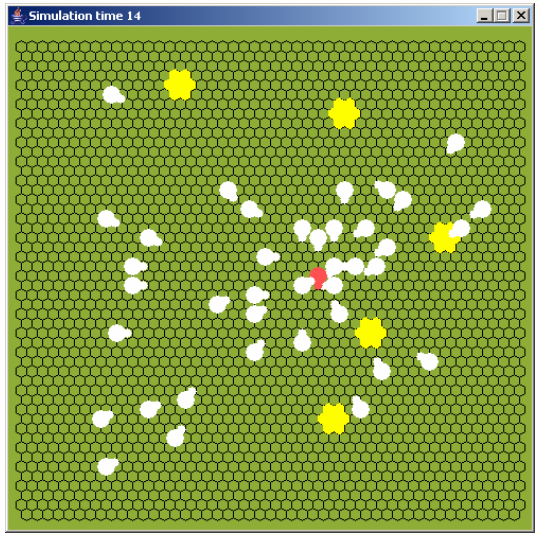
Emergence prediction



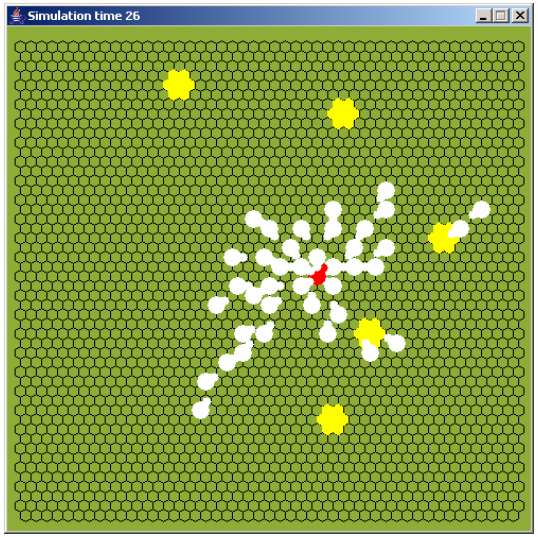
Time: 2



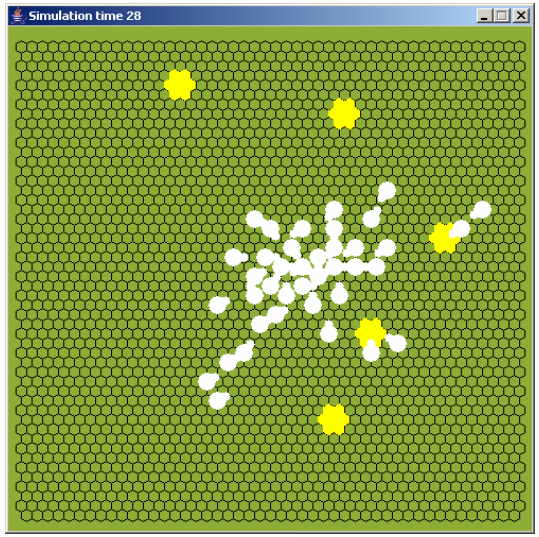
Time: 9



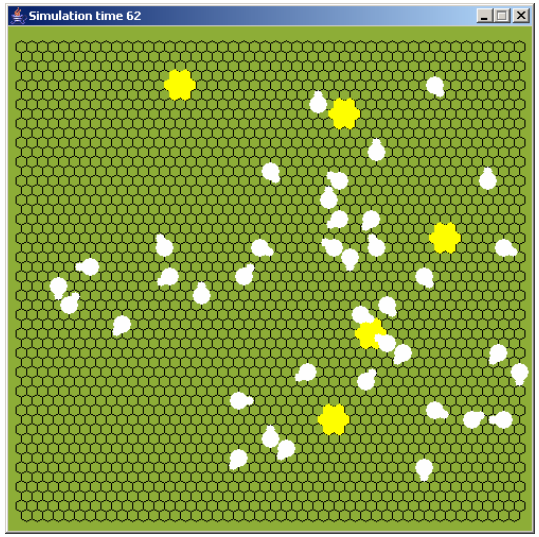
Time: 14



Time: 26

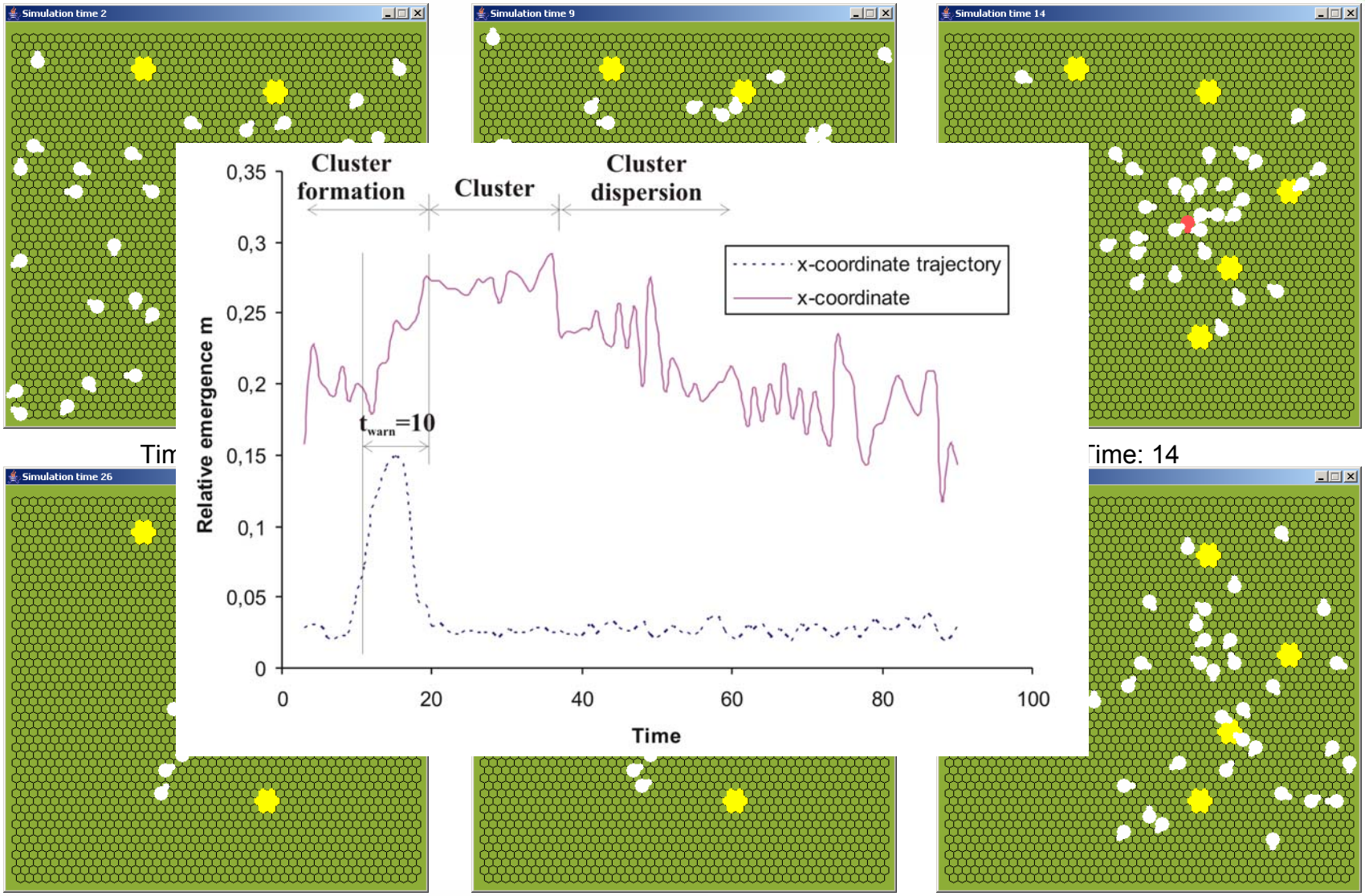


Time: 28



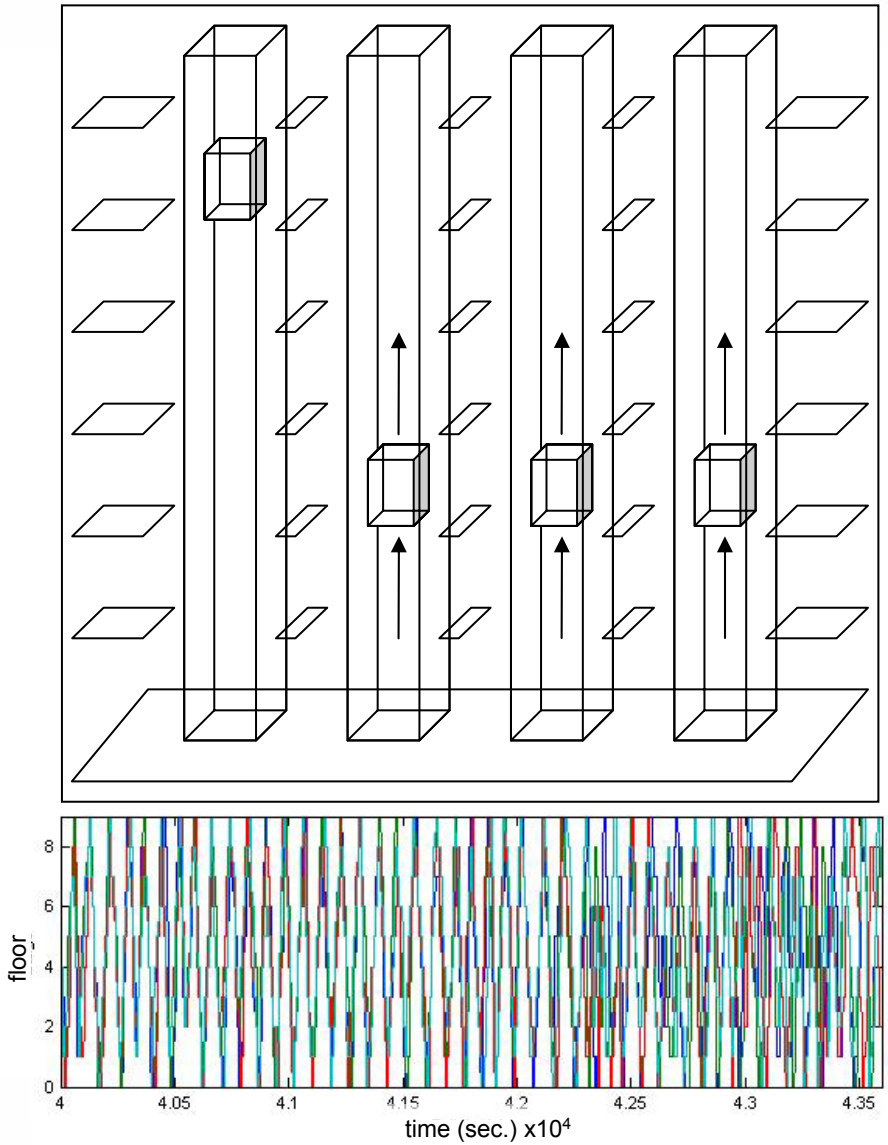
Time: 62

Emergence prediction



Elevator simulation

- Elevator group with inter-floor traffic
- Decentralised, self-organised behaviour using collective control
  - Every cabin always stops at the nearest hall call in its running direction
- Bunching effect (emergence)
  - Tend to synchronise (move up and down as a parallel wave)
  - Inefficient behaviour (substitute with one huge single elevator)



Summary and outlook

- Summary
  - Implementation of 2 test scenarios
  - Definition of emergence as self-organised order
  - A generic observer/controller architecture
  - First experimental results

- Outlook
  - Focussing on technical scenarios (robot swarms, elevator, …).
  - Integration of machine learning into the observer/controller loop.
  - Optimisation of the observer/controller architecture.
  - Closing the loop.

Publications

- Branke, J., Mnif, M., Müller-Schloer, C., Prothmann, H., Richter, U., Rochner, F., and Schmeck, H.: Organic computing – addressing complexity by controlled self-organization. To appear in Proceedings of ISOLA 2006.
- Mnif, M., Müller-Schloer, C.: Quantitative emergence. In: Proceedings of the 2006 IEEE MountainWorkshop on Adaptive and Learning Systems (IEEE SMCals 2006). (2006) 78–84
- Richter, U., Mnif, M., Branke, J., Müller-Schloer, C., Schmeck, H.: Towards a generic observer/controller architecture for organic computing. In Hochberger, C., Liskowsky, R., eds.: INFORMATIK 2006 – Informatik für Menschen! Volume P-93 of GI-Edition – Lecture Notes in Informatics (LNI)., Bonner Köllen Verlag (2006) 112–119
- Schmeck, H.: Organic Computing - A new vision for distributed embedded systems. In: Proceedings of the 8th IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC 2005), IEEE Computer Society (2005) 201–203
- Müller-Schloer, C.: Organic Computing: On the feasibility of controlled emergence. In Orailoglu, A., Chou, P.H., Eles, P., Jantsch, A., eds.: Proceedings of the 2nd IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS 2004), ACM (2004) 2–5