

LECTURE 7: REACTIVE AND HYBRID AGENTS

An Introduction to Multiagent Systems

CIS 716.5, Spring 2005

Reactive Architectures

- There are many unsolved (some would say insoluble) problems associated with symbolic AI.
- These problems have led some researchers to question the viability of the whole paradigm, and to the development of *reactive* architectures.
- Although united by a belief that the assumptions underpinning mainstream AI are in some sense wrong, reactive agent researchers use many different techniques.
- In this presentation, we start by reviewing the work of one of the most vocal critics of mainstream AI: Rodney Brooks.

Brooks — behaviour languages

- Brooks has put forward three theses:
 1. Intelligent behaviour can be generated *without* explicit representations of the kind that symbolic AI proposes.
 2. Intelligent behaviour can be generated *without* explicit abstract reasoning of the kind that symbolic AI proposes.
 3. Intelligence is an *emergent* property of certain complex systems.

- He identifies two key ideas that have informed his research:
 1. Situatedness and embodiment: ‘Real’ intelligence is situated in the world, not in disembodied systems such as theorem provers or expert systems.
 2. Intelligence and emergence: ‘Intelligent’ behaviour arises as a result of an agent’s interaction with its environment. Also, intelligence is ‘in the eye of the beholder’; it is not an innate, isolated property.

- To illustrate his ideas, Brooks built some based on his *subsumption architecture*.
- A subsumption architecture is a hierarchy of task-accomplishing *behaviours*.
- Each behaviour is a rather simple rule-like structure.
- Each behaviour 'competes' with others to exercise control over the agent.
- Lower layers represent more primitive kinds of behaviour, (such as avoiding obstacles), and have precedence over layers further up the hierarchy.

- The resulting systems are, in terms of the amount of computation they do, *extremely* simple.
- Some of the robots do tasks that would be impressive if they were accomplished by symbolic AI systems.
- Steels' Mars explorer system, using the subsumption architecture, achieves near-optimal cooperative performance in simulated 'rock gathering on Mars' domain:

The objective is to explore a distant planet, and in particular, to collect sample of a precious rock. The location of the samples is not known in advance, but it is known that they tend to be clustered.

- For individual (non-cooperative) agents, the lowest-level behavior, (and hence the behavior with the highest “priority”) is obstacle avoidance:

if detect an obstacle *then* change direction. (1)

- Any samples carried by agents are dropped back at the mother-ship:

if carrying samples *and* at the base *then* drop samples (2)

- Agents carrying samples will return to the mother-ship:

if carrying samples *and not* at the base *then* travel up gradient. (3)

- Agents will collect samples they find:

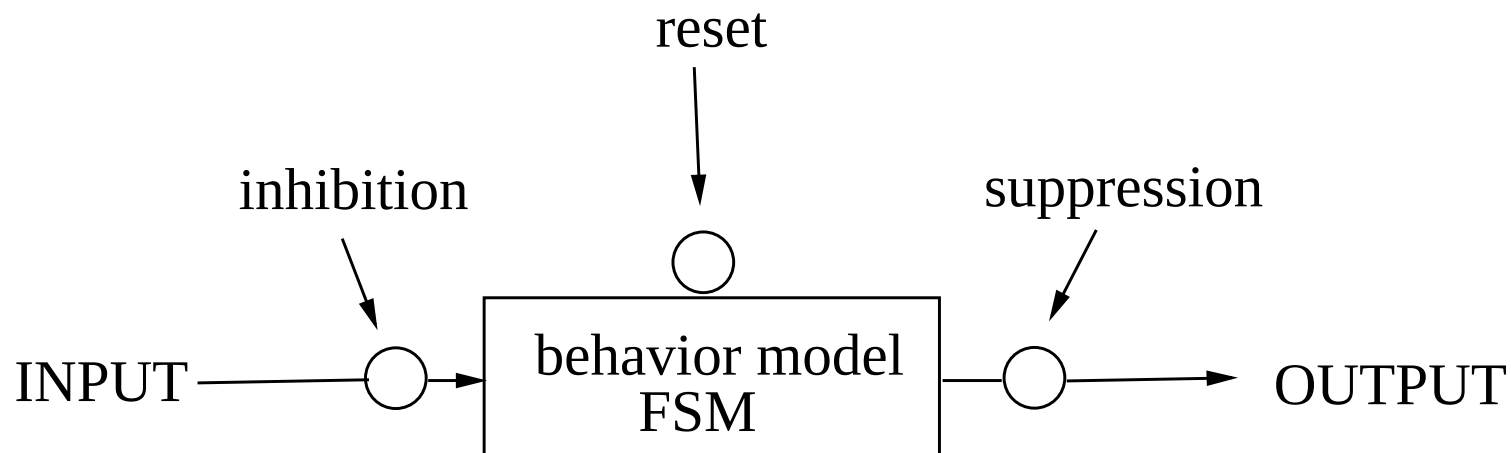
if detect a sample *then* pick sample up. (4)

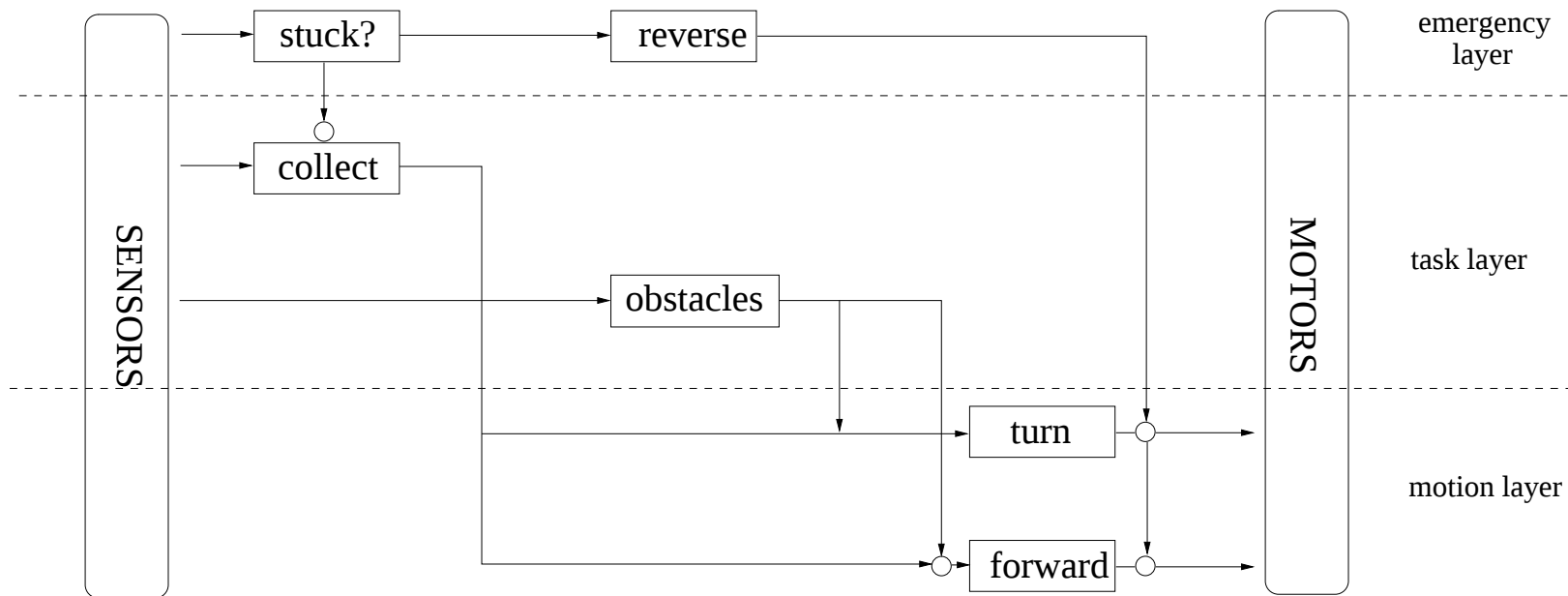
- An agent with “nothing better to do” will explore randomly:

if true *then* move randomly. (5)

Abstract view of subsumption architecture

- Layered approach based on levels of competence
- Augmented finite state machine:





Situated Automata

- A sophisticated approach is that of Rosenschein and Kaelbling.
- In their *situated automata* paradigm, an agent is specified in a rule-like (declarative) language, and this specification is then compiled down to a digital machine, which satisfies the declarative specification.

This digital machine can operate in a *provable time bound*.

- Reasoning is done *off line*, at *compile time*, rather than *online* at *run time*.

- The theoretical limitations of the approach are not well understood.
- Compilation (with propositional specifications) is equivalent to an NP-complete problem.
- The more expressive the agent specification language, the harder it is to compile it.

(There are some deep theoretical results which say that after a certain expressiveness, the compilation simply can't be done.)

Emergent behaviour

- Important but not well-understood phenomenon
- Often found in behaviour-based/reactive systems
- Agent behaviours “emerge” from
 - Interactions of rules
 - Interactions of behaviours
 - Interactions of either with environment

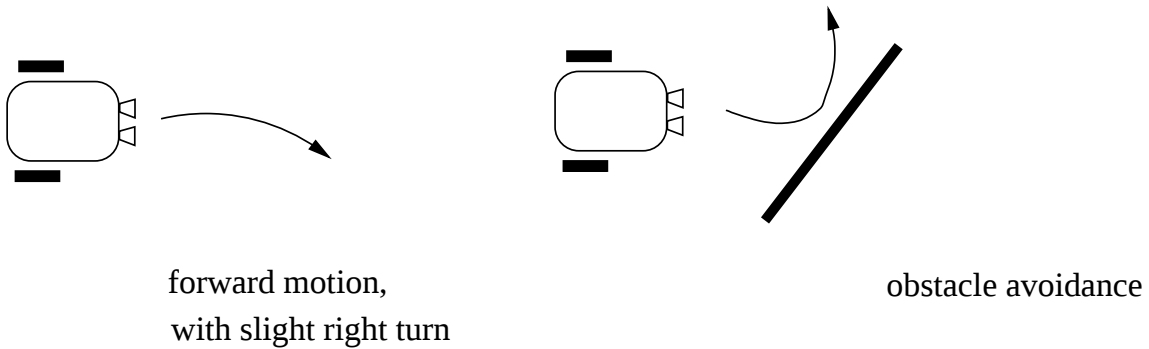
- Coded behaviour
 - In the programming scheme
- Observed behaviour
 - In the eyes of the observer
- There is no one-to-one mapping between the two!
- When Observed behaviour “exceeds” programmed behaviour, then we have emergence.

- Is it magic?
 - Sum is greater than the parts
 - Emergent behaviour is more than the controller that produces it
- Interaction and emergence.
 - Interactions between rules, behaviours and environment
 - Source of expressive power for a designer
 - Systems can be designed to take advantage of emergent behaviour

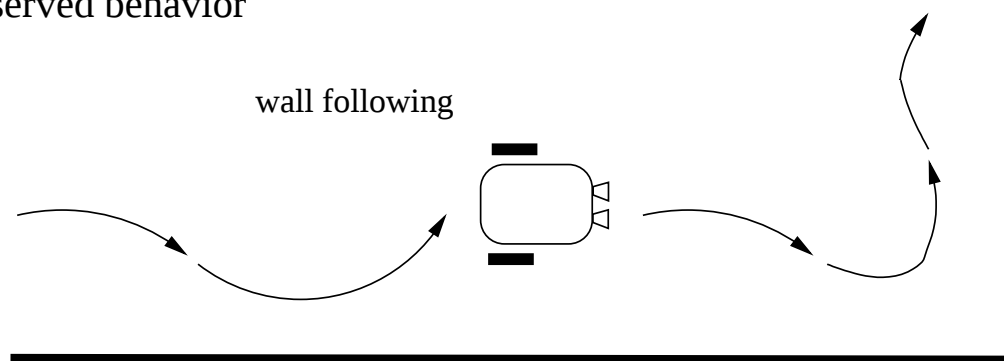
- Emergent flocking.
- Program multiple agents:
 - Don't run into any other robot
 - Don't get too far from other robots
 - Keep moving if you can
- When run in parallel on many agents, the result is flocking

Wall following

coded behavior



observed behavior



- Can also be implemented with these rules:
 - If too far, move closer
 - If too close, move away
 - Otherwise, keep on
- Over time, in an environment with walls, this will result in wall-following
- Is this emergent behaviour?

- Can argued yes because
 - Robot itself is not aware of a wall, it only reacts to distance readings
 - Concepts of “wall” and “following” are not stored in the robot’s controller
 - The system is just a collection of rules
- Notion of emergence depends on two aspects:
 - Existence of an external observer, to observe and describe the behaviour of the system
 - Access to the internals of the controller itself, to verify that the behaviour is not explicitly specified anywhere in the system

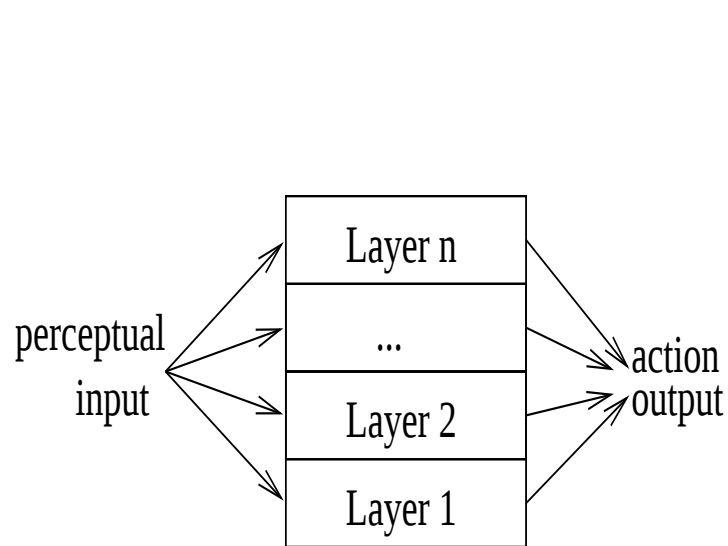
- Unexpected vs emergent.
 - Some researchers say the above is not enough for behaviour to be emergent, because above is programmed into the system and the “emergence” is a matter of semantics
 - So emergence must imply something unexpected, something “surreptitiously discovered” by observing the system.
 - “Unexpected” is highly subjective, because it depends on what the observer was expecting
 - Naïve observers are often surprised!
 - Informed observers are rarely surprised
- Once a behaviour is observed, it is no longer unexpected
- Is new behaviour then “predictable”?

Hybrid Architectures

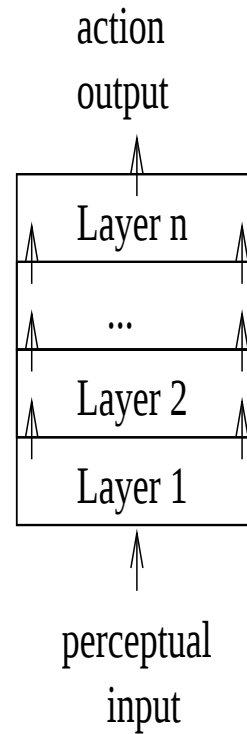
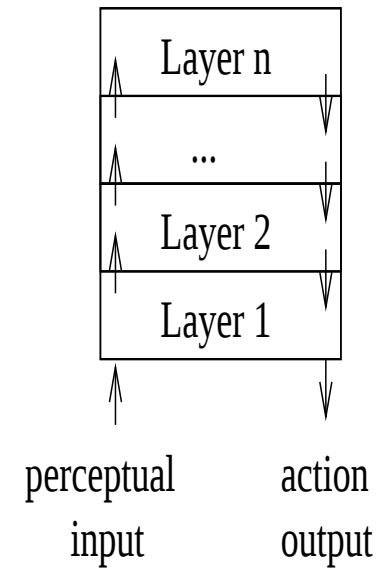
- Many researchers have argued that neither a completely deliberative nor completely reactive approach is suitable for building agents.
- They have suggested using *hybrid* systems, which attempt to marry classical and alternative approaches.
- An obvious approach is to build an agent out of two (or more) subsystems:
 - a *deliberative* one, containing a symbolic world model, which develops plans and makes decisions in the way proposed by symbolic AI; and
 - a *reactive* one, which is capable of reacting to events without complex reasoning.

- Often, the reactive component is given some kind of precedence over the deliberative one.
- This kind of structuring leads naturally to the idea of a *layered* architecture, of which TOURINGMACHINES and INTERRAP are examples.
- In such an architecture, an agent's control subsystems are arranged into a hierarchy, with higher layers dealing with information at increasing levels of abstraction.

- A key problem in such architectures is what kind control framework to embed the agent's subsystems in, to manage the interactions between the various layers.
- *Horizontal layering.*
Layers are each directly connected to the sensory input and action output.
In effect, each layer itself acts like an agent, producing suggestions as to what action to perform.
- *Vertical layering.*
Sensory input and action output are each dealt with by at most one layer each.

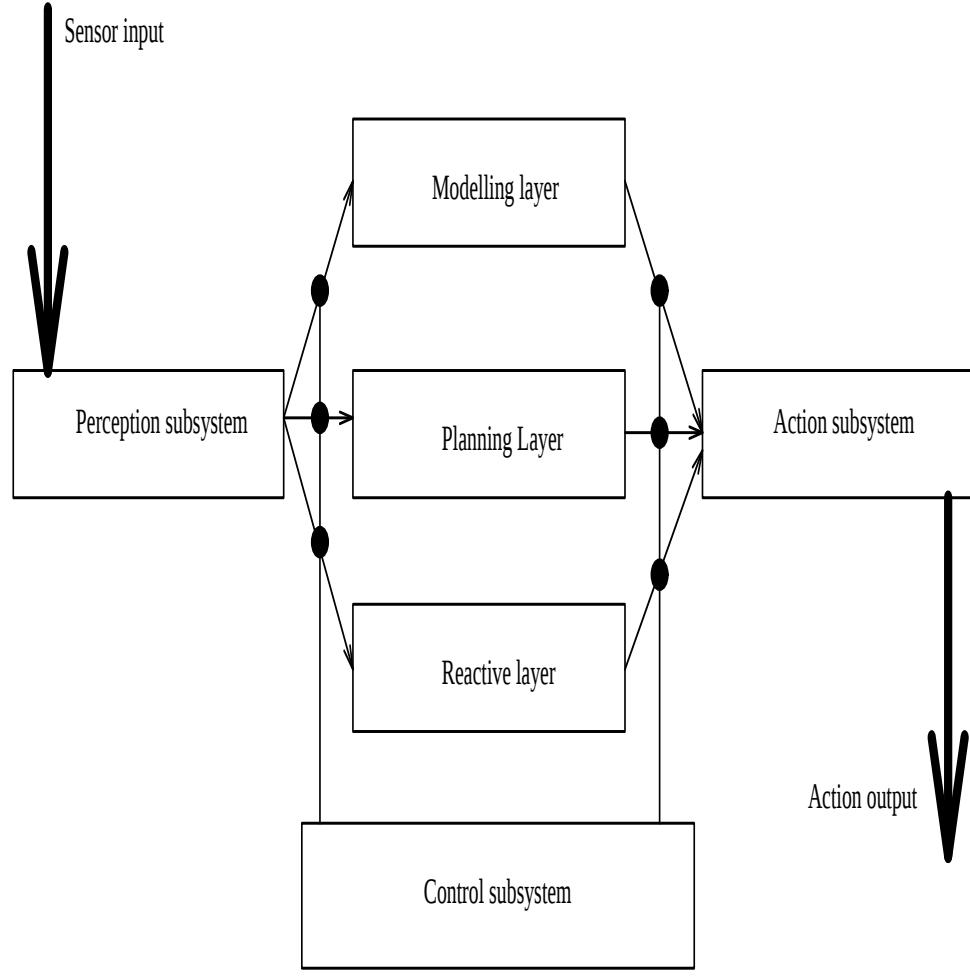


(a) Horizontal layering

(b) Vertical layering
(One pass control)(c) Vertical layering
(Two pass control)

Ferguson — TOURINGMACHINES

- The TOURINGMACHINES architecture consists of *perception* and *action* subsystems, which interface directly with the agent's environment, and three *control layers*, embedded in a *control framework*, which mediates between the layers.



- The *reactive layer* is implemented as a set of situation-action rules, à la subsumption architecture.

Example:

```
rule-1: kerb-avoidance
  if
    is-in-front(Kerb, Observer) and
    speed(Observer) > 0 and
    separation(Kerb, Observer) < KerbThreshold
  then
    change-orientation(KerbAvoidanceAngle)
```

- The *planning layer* constructs plans and selects actions to execute in order to achieve the agent's goals.

- The *modelling layer* contains symbolic representations of the ‘cognitive state’ of other entities in the agent’s environment.
- The three layers communicate with each other and are embedded in a control framework, which use *control rules*.

Example:

```
sensor-rule-1:  
  if  
    entity(obstacle-6) in perception-buffer  
  then  
    remove-sensory-record(layer-R, entity(obstacle-6))
```

- Such control structures have become common in robotics.

Summary

- This lecture has looked at two further kinds of agent:
 - Reactive agents; and
 - Hybrid agents.
- Reactive agents build complex behaviour from simple components.
- Complex to build complex agents.
- Hybrid agents try to combine the speed of reactive agents with the power of deliberative agents.
- Hybrid agents are common in robotics.