BEHAVIOUR-BASED ROBOTICS

Overview

- Control
- Behaviour-based systems
- Expressing behaviours
- Behavioural encoding
- Representations
- Behaviour coordination
- Emergent behaviour

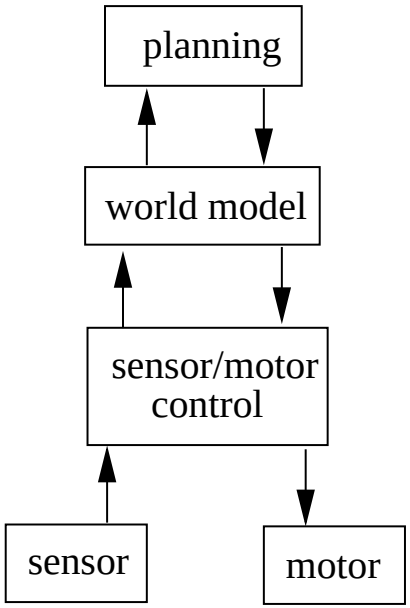
Control: models

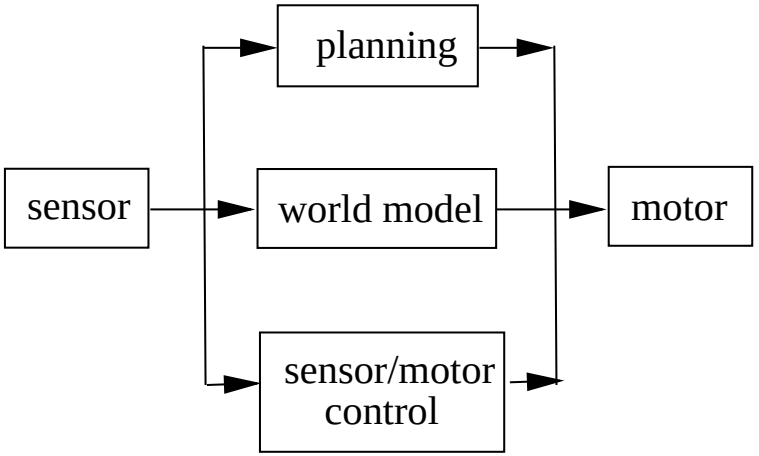
- We like to make a distinction between:
  - Classic "model-based" AI
    - symbolic representations
  - Neo "behaviour-based" AI
    - numeric representations
- Classic models are "good old-fashioned AI", in the tradition of McCarthy.
- Behaviour-based models are "nouvelle AI" in the tradition of Brooks.
- (There are also hybrid models that combine aspects of both model-based and behaviour-based, but we have no time to cover them in detail here.)

Control: classic models

- Deliberative... sense, plan, act
  - functional decomposition
  - systems consist of sequential modules achieving independent functions
    - sense world
    - generate plan
    - translate plan into actions
- Reactive architectures
  - task-oriented decomposition
  - systems consist of concurrently executed modules achieving specific tasks
    - avoid obstacle
    - follow wall

Control: two orthogonal flows





Control: behaviour based systems

- Behaviours are the underlying module of the system
- Behavioural decomposition
  - rather than a functional or a task-oriented decomposition
- Systems consist of sequential modules achieving independent functions
- Natural fit to robotic behaviour
  - generate a motor response from a given perceptual stimulus
  - basis in biological studies
  - biology is an inspiration for design
- Abstract representation is avoided

Behaviour based systems: behaviour vs action

Behaviour is:

- based on dynamic processes
  - operating in parallel
  - lack of central control
  - fast couplings between sensors and motors
- exploiting emergence
  - side-effects from combined processes
  - using properties of the environment
- reactive

Behaviour-based systems: behaviour vs action

Action is:

- discrete in time
  - well defined start and end points
  - allows pre- and post-conditions
- avoidance of side-effects
  - only one (or a few) actions at a time
  - conflicts are undesired and avoided
- deliberative

Actions are building blocks for behaviours.

Behaviour-based systems: properties

- Achieve specific tasks/goals
  - avoid others, find friend, go home
- Typically execute concurrently
- Can store state and be used to construct world models/representations
- Can directly connect sensors to effectors
- Can take inputs from other behaviours and send outputs to other behaviours
  - connection in networks
- Typically higher-level than actions (go home, not turn left 45 degrees)

Behaviour-based systems: key properties

- Ability to act in real time
- Ability to use representations to generate efficient (not only reactive) behaviour
- Ability to use a uniform structure and representation throughout the system (so no intermediate layer)

Behaviour-based systems: challenges

- How can representation be effectively distributed over the behaviour structure?
  - time scale must be similar to that of real-time components of the system
  - representation must use same underlying behaviour structure for all components of the system
- Some components may be reactive
- Not every component is involved with representational computation
- Some systems use a simple representation
- As long as the basis is in behaviours and not rules, the system is a BBS

Behaviour-based systems: what are behaviours?

- Behaviour: anything observable that the system/robot does
  - how do we distinguish internal behaviours (components of a BBS) and externally observable behaviours?
  - should we distinguish?
- Reactive robots display desired external behaviours
  - avoiding
  - collecting cans
  - walking
- But controller consists of a collection of rules, possibly in layers
- BBS actually consist and are programmed in the behaviours, which are higher granularity, extended in time, capable of representation

Behaviour-based systems: expressing behaviours

- Behaviours can be expressed with various representations
- When a control system is being designed, the task is broken down into desired external behaviours
- Those can be expressed with
  - Functional notation
  - Stimulus response (SR) diagrams
  - Finite state machines/automata (FSA)
  - Schema

Expressing behaviours: functional notation

- Mathematical model – represented as triples (S, R, β)

  S = stimulus

  R = range of response

  β = behavioural mapping between S and R

- Easily convert to functional languages like LISP

```
coordinate-behaviours [
    move-to-classroom ( detect-classroom-location ),
    avoid-objects ( detect-objects ),
    dodge-students ( detect-students ),
    stay-to-right-on-path ( detect-path ),
    defer-to-elders ( detect-elders )
] = motor-response
```

Expressing behaviours: FSA diagrams

- States of the diagram can also be called behaviours, diagrams show sequences of behaviour transitions
- Situated automata.
  - formalism for specifying FSAs that are situated
  - task described in high-level logic expressions, as a set of goals and a set of operators that achieve (ach) and maintain (maint) the goals
  - once defined, tasks can be compiled into circuits (using special purpose languages), which are reactive

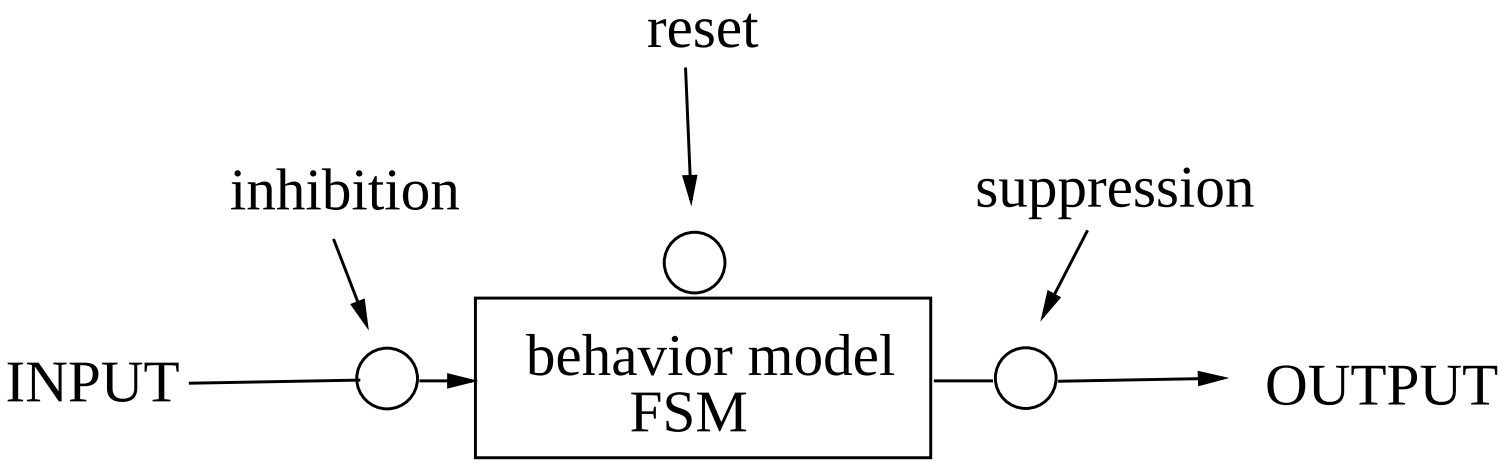
Expressing behaviours: FSA diagrams

- Example:

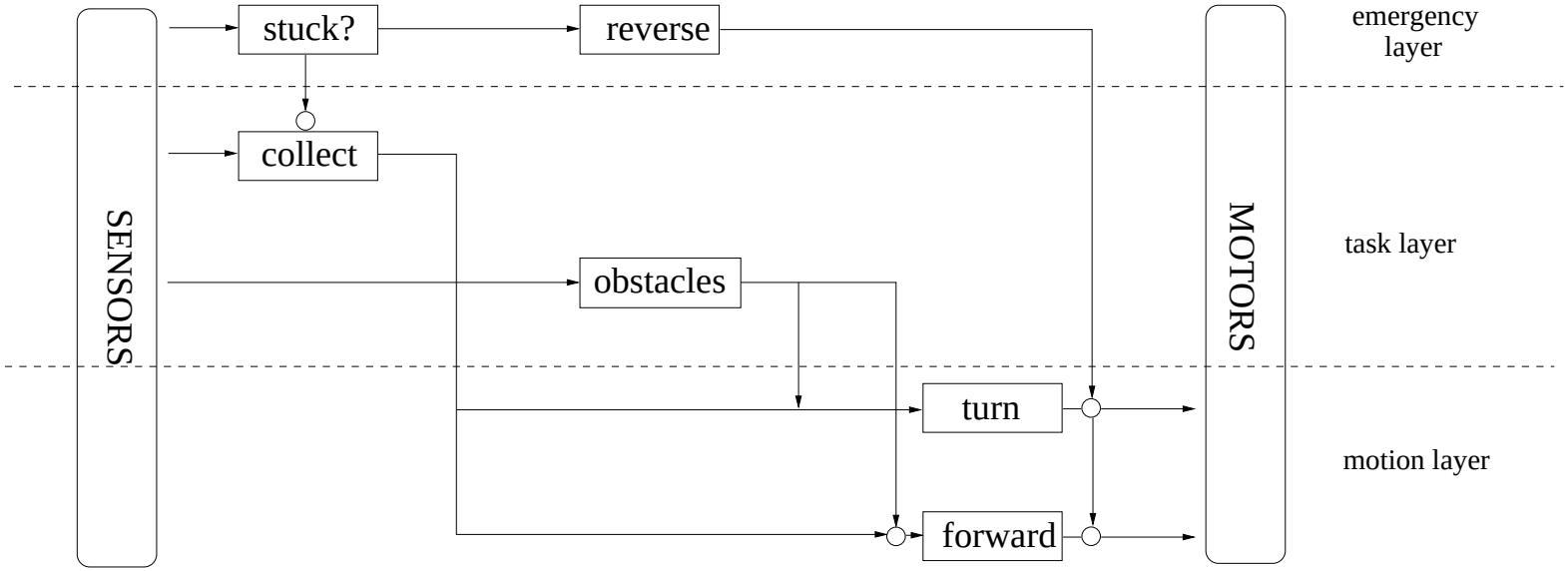
```
(defgoalr (ach in-classroom)
(if (not start-up)
    (maint (and (maint move-to-classroom)
                (maint avoid-objects)
                (maint dodge-students)
                (maint stay-to-right-on-path)
                (maint defer-to-elders)))))
```

Expressing behaviours: subsumption architecture

- Rodney Brooks, 1986, MIT AI lab
- Reactive elements
- Behaviour-based elements
- Layered approach based on levels of competence
- Augmented finite state machine:



Expressing behaviours: subsumption architecture



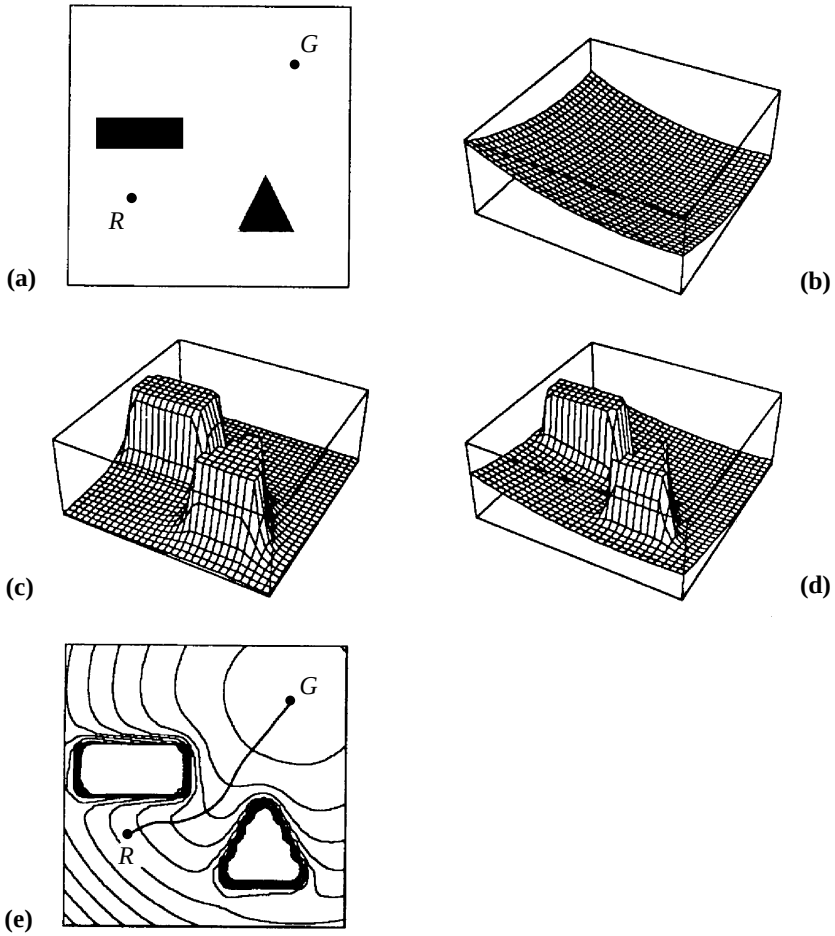
Expressing behaviours: formally

- Behavioural response in physical space has a strength and an orientation
- Expressed as (S, R, β)
- S = stimulus, necessary but not sufficient condition to evoke a response (R); internal state can also be used
- β = behavioural mapping categories
  - null
  - discrete
  - continuous

Expressing behaviours: behavioural mapping

- Discrete encoding
  - expressed as a finite set of situation-response pairs/mappings
  - mappings often include rule-based form IF-THEN
  - examples:
    - Gapps [Kaelbling & Rosenschein]
    - subsumption language [Brooks]
- Continuous encoding
  - instead of discretizing the input and output, a continuous mathematical function describes the input-output mapping
  - can be simple, time-varying, harmonic
  - example: potential field

Expressing behaviours: potential field



Behavioural encoding: strengths and weaknesses

- Strengths
  - support for parallelism
  - run-time flexibility
  - timeliness for development
  - support for modularity
- Weaknesses
  - niche targetability
  - hardware retargetability
  - combination pitfalls (local minima, oscillations)

Behaviour coordination

- BBS consist of collection of behaviours
- Execution must be coordinated in a consistent fashion
- Coordination can be
  - competitive
  - cooperative
  - combination of the two
- Deciding what to do next.
  - action-selection problem
  - behaviour-arbitration problem

Behaviour coordination

- Competitive coordination.
  - perform arbitration (selecting one behaviour among a set of candidates)
    - priority-based: subsumption
    - state-based: discrete event systems
    - function-based: spreading of activation action selection
- Cooperative coordination.
  - perform command fusion (combine outputs of multiple behaviours)
  - voting
  - fuzzy (formalized voting)
  - superposition (linear combinations) i.e. potential fields.

Emergent behaviour: what is it?

- Important but not well-understood phenomenon
- Robot 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
  - emergence
- There is no one-to-one mapping between the two!

Emergent behaviour: how does it arise?

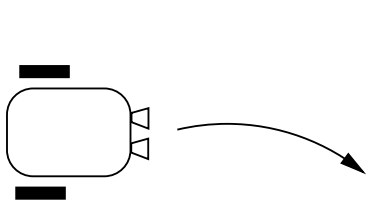
- Is it magic?
  - sum is greater than the parts
  - emergent behaviour is more than the controller that produces it
- Interaction and emergence.
  - interactions between rules, behaviours and environment
  - source of expressive power for a designer
  - systems can be designed to take advantage of emergent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

Emergent behaviour: an example

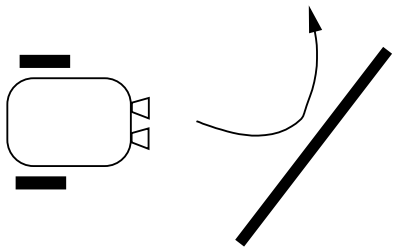
- Emergent flocking.
- Program multiple robots:
  - don't run into any other robot
  - don't get too far from other robots
  - keep moving if you can
- When run in parallel on many robots, the result is flocking

Emergent behaviour: wall following

coded behavior



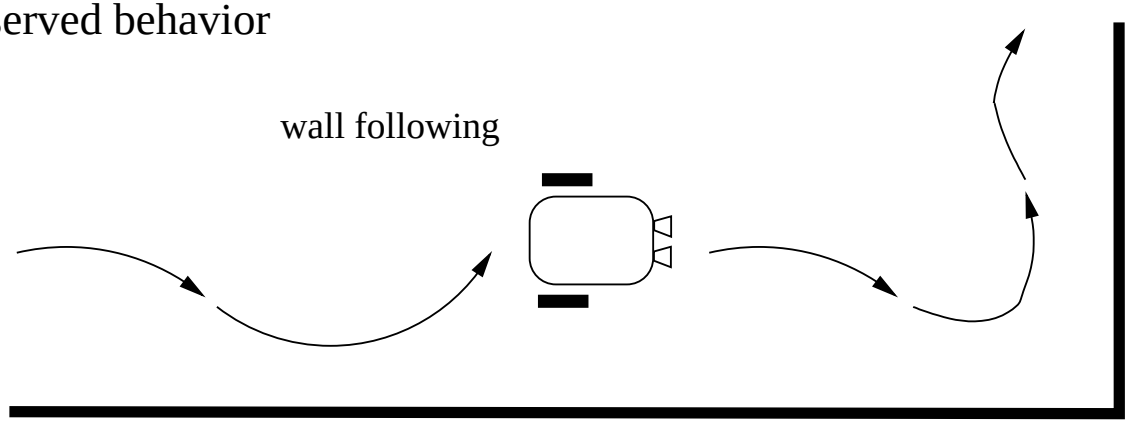
forward motion,
with slight right turn



obstacle avoidance

observed behavior

wall following



Emergent behaviour: wall following

- 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?
- It is 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

Emergent behaviour: conditions on emergence

- 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

Emergent behaviour: conditions on emergence

- 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"?