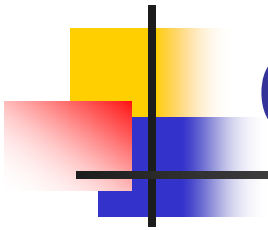
COMP5121
Mobile Robots

Architectures and Behaviours

Dr. Mario Gongora
mgongora@dmu.ac.uk



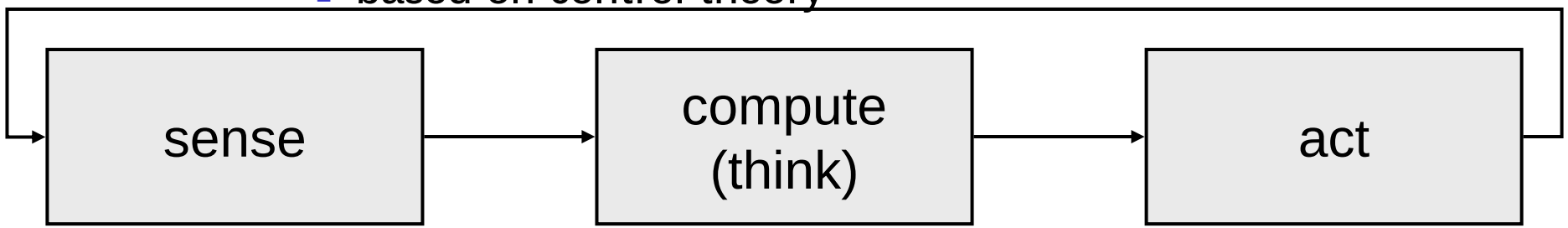
Overview

- Control models
  - the *sense-think-act* control cycle
  - model-based controllers
- Reactive robotics
  - reactive controllers
  - behaviour-based controllers
- Other approaches
  - other reactive controllers
  - hybrid controllers
  - learning robots

Control Models

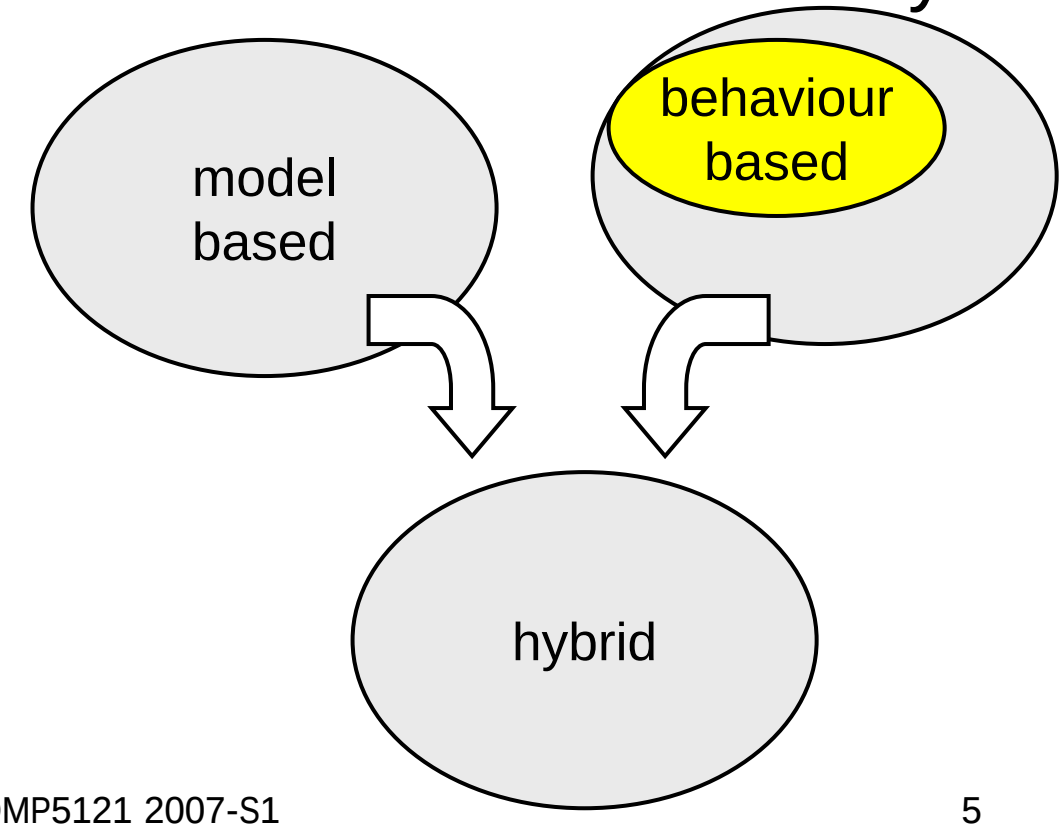
The Control Cycle

- A fundamental methodology derived in the early days of robotics from engineering principles is the *sense-think-act* cycle
  - the principle is to continuously attempt to minimise the error between the actual state and the desired state
    - based on control theory



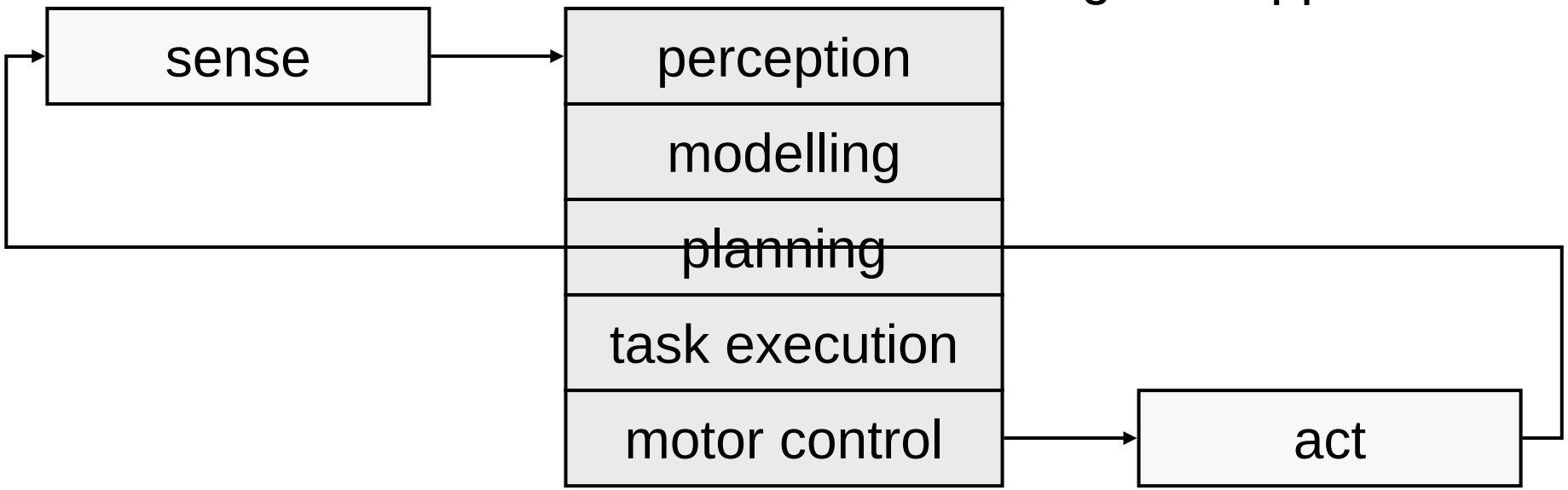
Control Architectures

- A variety of different approaches have been tried for implementing the *sense-think-act* control cycle
- These approaches can be categorised as
  - model-based
  - reactive
  - hybrid

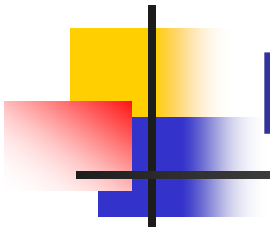


Model Based

- A symbolic internal ‘world-model’ is maintained
  - the sub-tasks are *decomposed* into functional layers
  - similar to ‘classical’ artificial intelligence approach

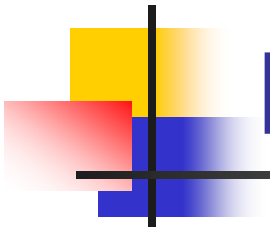


Problems with Models



- An adequate, accurate and up-to-date model must be maintained at all times
  - this is very difficult in practice!
  - what if sensors detect an object that hasn't been defined
- A model-based system is extremely brittle
  - if one of the functional layers fails (e.g. hardware problems, software bugs), then the whole system fails
- Significant processing power is required
  - maintaining the model takes time, so slow responses!?
- Despite much effort, little progress was made!

Reactive Robotics

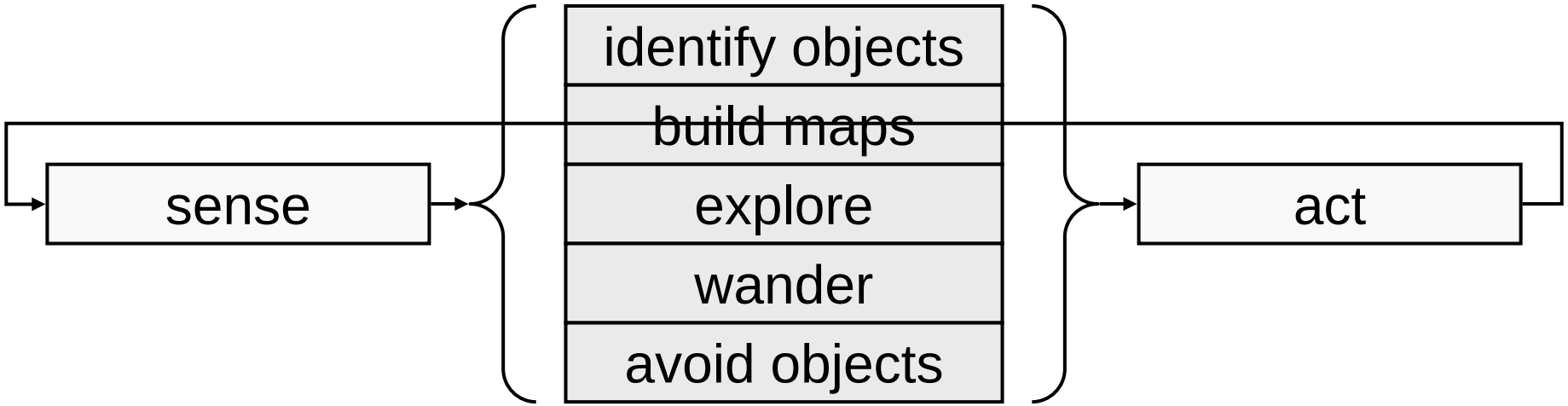


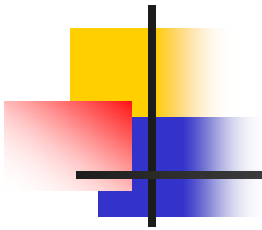
Reactive Controllers

- In order to try to overcome the shortcomings of model-based robots, modern approaches have centred predominantly on simple *reactive* systems with minimal amounts of computation
  - 'model-free systems'
- More correctly, the models are *simple* and *implicit*
  - the systems do not use symbolic models but, for example, a rule-set which tells a robot how to react to a corner when following a wall may be considered to be a simple, implicit model fragment
    - it implicitly encodes assumptions about the environment

Behaviour Based

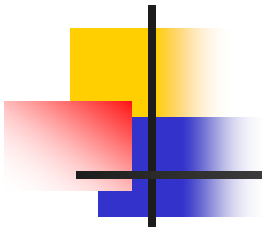
- The control system is broken down into horizontal modules, or *behaviours*, that run in parallel
  - each behaviour has direct access to sensor readings and can control the robot's motors directly





Behaviour Advantages

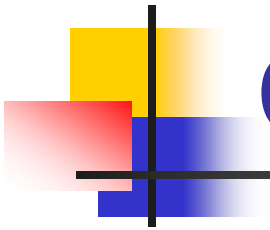
- It supports multiple goals and is more efficient
  - there is no functional hierarchy between layers
    - one layer does not call another layer
  - each layer can work on different goals in parallel
  - communication between layers is achieved via message passing which need not be synchronised
- The system is easier to design, debug and extend
  - each module can be designed and tested individually
- The system is robust
  - if one module fails, e.g. *wander*, then other layers, e.g. *avoid obstacles*, still function and behave correctly



Behaviour Limitations

- It is extremely difficult to implement plans
  - in pure form a behaviour-based robot has no memory (not even an internal state memory) and so is unable to follow an externally specified sequences of actions
- It can be very hard to predict how a large number of multiple behaviours may interact
  - *emergent behaviour* is the term given to unexpected behaviour that comes about through these interactions
    - sometimes it is useful, sometimes it is not!
- The robot can get trapped in a *limit cycle*
  - trapped in a dead-end, repeatedly turning left then right

Other Approaches

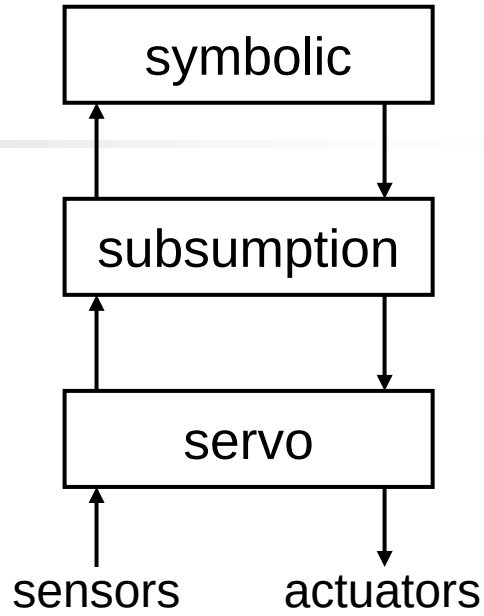


Other Reactive Approaches

- Two other reactive approaches that are popular
- Potential field methods
  - a potential field is a concept from physics
    - e.g. the *gravitational field*
      - you do not need to be told which way to fall
      - planets do not need to plan how to move around the sun
    - obstacles exert hypothetical repulsive forces on the robot
- Motor schema navigation
  - multiple, concurrent schema generate separate behaviours which are summed to produce output
    - schema are dynamically created/destroyed as needed

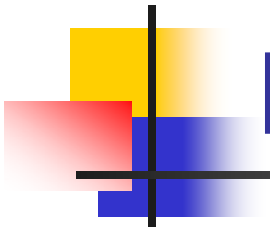
Hybrid Approaches

- The *SSS* three-layer architecture
  - the *servo-subsumption-symbolic* architecture combines Brooks' architecture with a lower-level servo control level and a higher-level symbolic system [Connell]



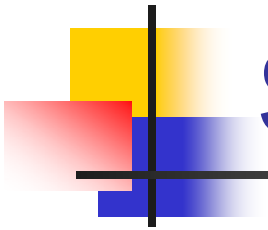
Fuzzy logic and neural network controllers

fuzzy logic rule-base(s), neural network(s) and combinations of both take inputs from sensors and process the data to generate output to actuators



Learning Approaches

- Traditional learning techniques
  - rather than attempt to predefine and predict a symbolic model of the ‘real-world’, the robot learns how to operate and how to behave by
    - supervised learning
      - desired output is known for each set of input settings (e.g. ANN’s)
    - reinforcement learning
      - learning by trial and error through performance feedback
- Evolutionary algorithms
  - using genetic algorithms to find good network weights
  - significant problems with evolving real solutions in reasonable time on current mobile robot hardware



Summary

- Summary of this lecture
  - control models
    - the *sense-think-act* control cycle
    - model-based controllers
  - reactive robotics
    - behaviour-based controllers
  - other approaches
    - other reactive controllers, hybrid controllers
    - learning robots
- Next lecture
  - hardware: sensors