Behavior-based Robot Design
An Introduction

Lecture #2, Sept 8, 2005

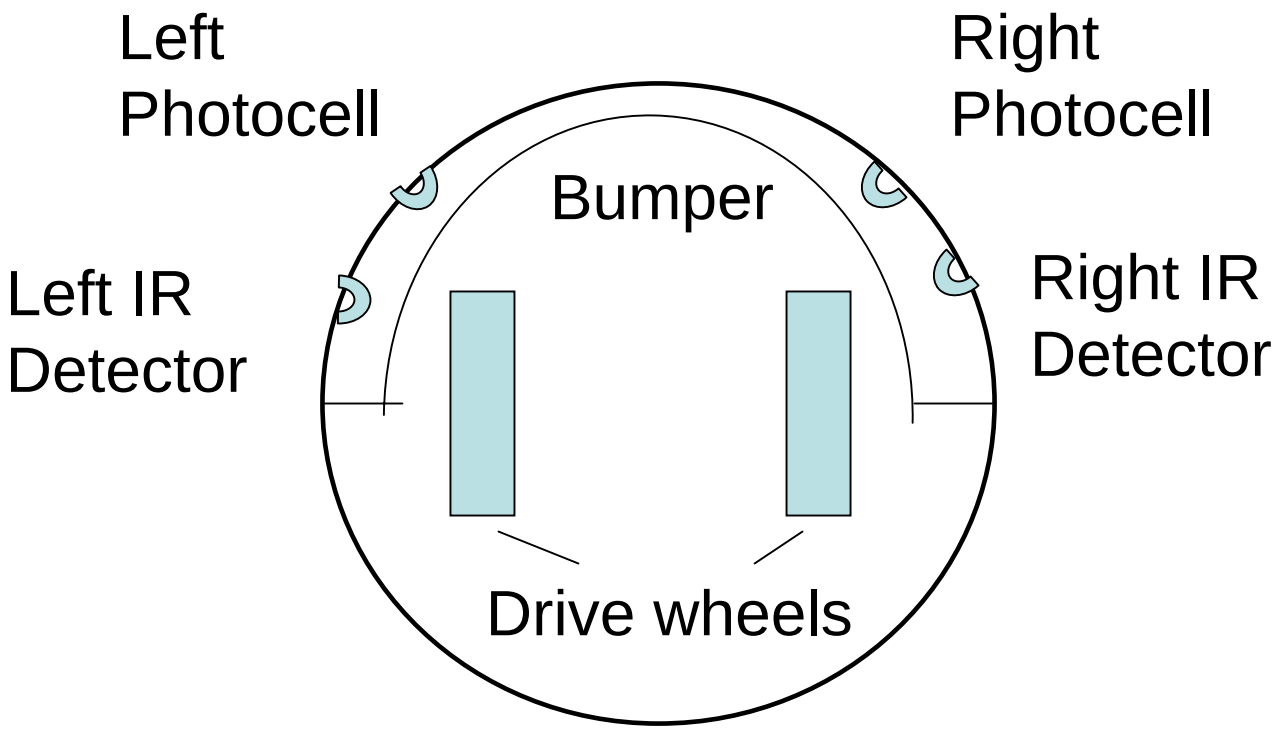
RSS II

Una-May O'Reilly

Agenda

I. Intuition of BB design with an example
II. Overview
III. Practicalities

I. A Collecting Robot in Simulation



www.behaviorbasedprogramming.com

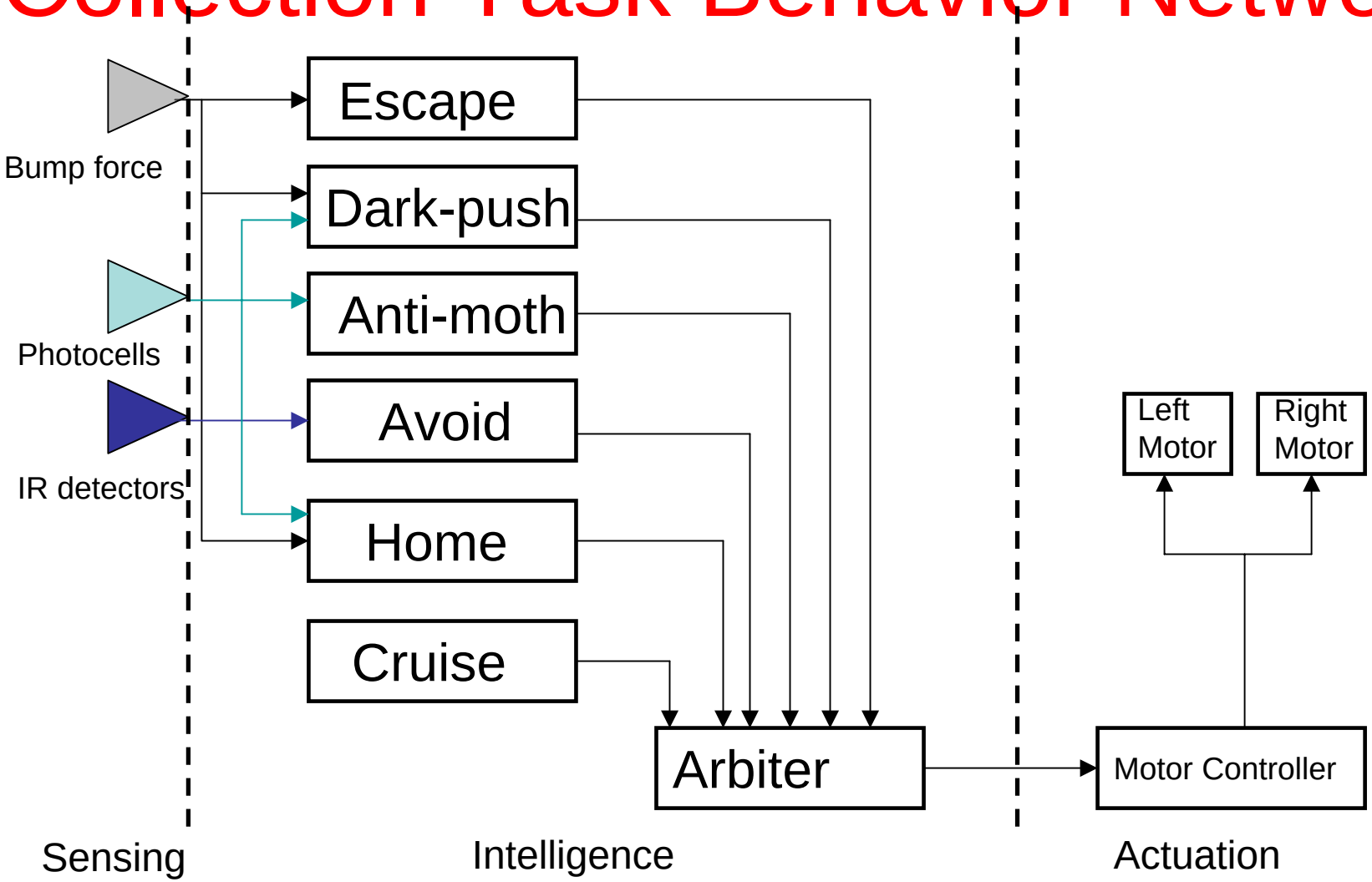
Questions

- What task is the robot doing?
  - Searching for pucks
  - When it finds one, pushes it to vicinity of the light source, goes to find another
  - Avoids or escapes from encounters with other objects

How is the robot collecting pucks?

- Task is decomposed as a set of simple behaviors (algorithms connecting sensors to actuation) that, when acting together, produce the overall activity

Collection Task Behavior Network



A Collecting Robot in Simulation

- The robots in BSim are circular differential drive robots with a bumper, two IR proximity sensors, two photo sensors and wheel encoders. The photo and IR sensors face diagonally from the front of the robot at 45 degree angles.
- Each robot supports a simple, yet powerful, behavior-based programming system which includes a set of primitive behaviors and a priority list arbiter.
- A robot's program is called a *task*. A task is a prioritized list of behaviors which all simultaneously compete to control the robot.
- The arbiter chooses which behavior is successful. You can program each robot by configuring a set of behaviors, prioritizing the behaviors for the arbiter, and then loading the behaviors into the robot.

Collection Behaviors

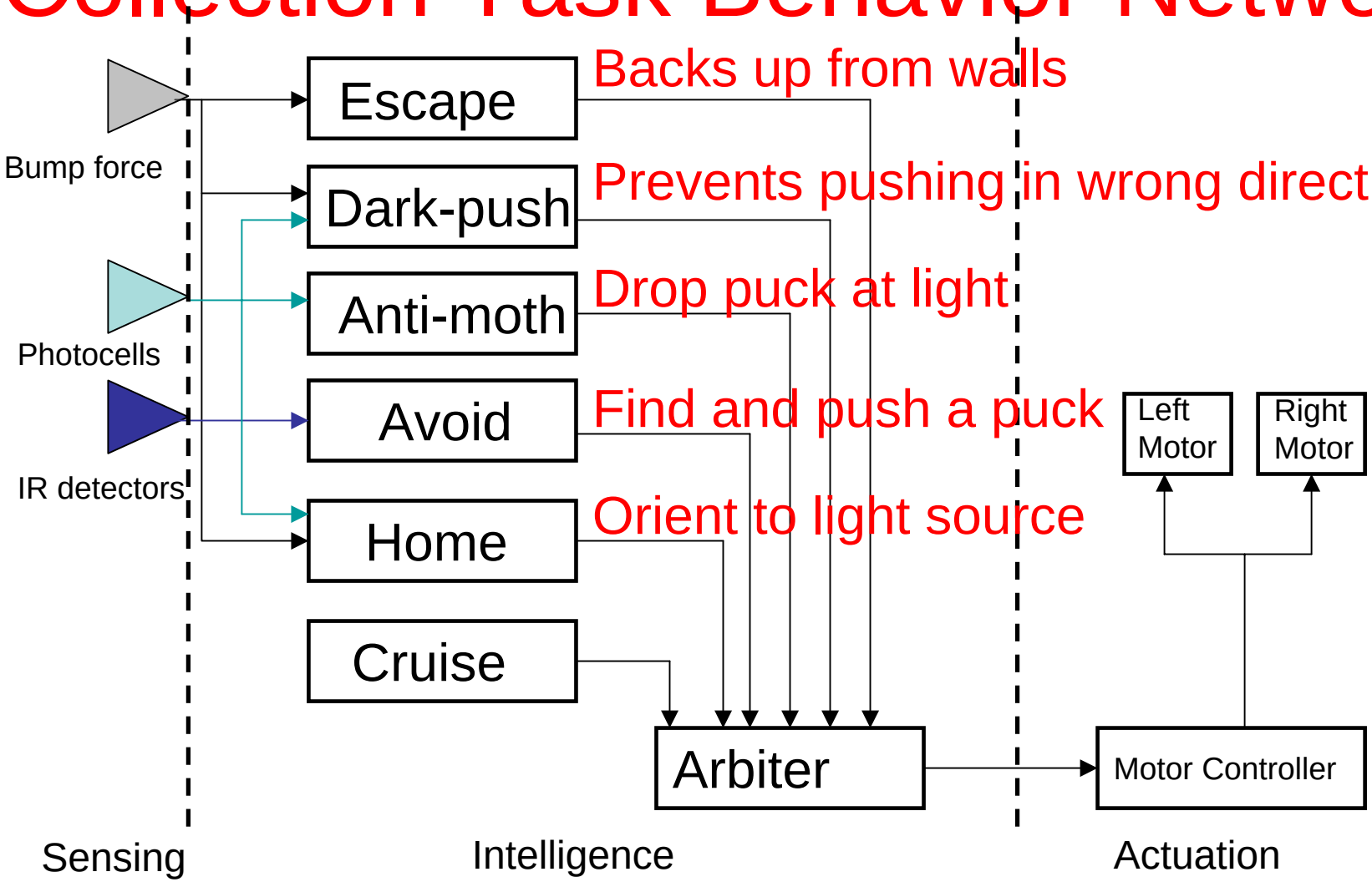
Cruise: drives the wheels at constant speeds. The behavior can try to drive the wheels at any speed, positive or negative, but the robot speed will max out at +/- 255.

- Home: tries to drive the robot toward a light source. It uses a proportional controller to home on a light source whenever the robot's photo sensors see light. The robot homes on the light by pivoting in the direction of the light and then moving forward a step. The robot determines the direction to the light by calculating the difference between the two photo sensor measurements..
- Avoid: Moves robot forward and left if the right proximity sensor is on, or forward and right is the left proximity sensor is on (if gain is positive). With a negative gain (in collection task) it goes toward an obstacle (eg a puck or wall)

Collection Behaviors

- Escape:a ballistic behavior triggered whenever the robot bumps into something. The behavior is performed in three steps: backup for a specified amount of time, spin a certain angle, and go forward for a specified amount of time.
- Anti-Moth: a ballistic behavior that triggers whenever the total light intensity measured by a photocell exceeds a threshold
- Dark-push: a ballistic behavior. It triggers whenever the robot tries to push something when no light is visible.

Collection Task Behavior Network



Things to Notice

- There’s no explicit FindPuck behavior
- No PushPuck behavior
- No DropPuck behavior
- These emerge from the interaction of the more primitive behaviors
- System behavior is not deterministic, but has random components
- Overall behavior is robust - ultimately collects pucks
- No representation of the world and no state

II. Overview: Artificial Creatures

- Contrast between good old fashioned Artificial Intelligence (GOFAI) and behavior-based AI
- GOFAI: Thought experiments on the nature of “intelligence” in creatures with bodies
- BB-AI draws inspiration from neurobiology, ethology, psychophysics, and sociology

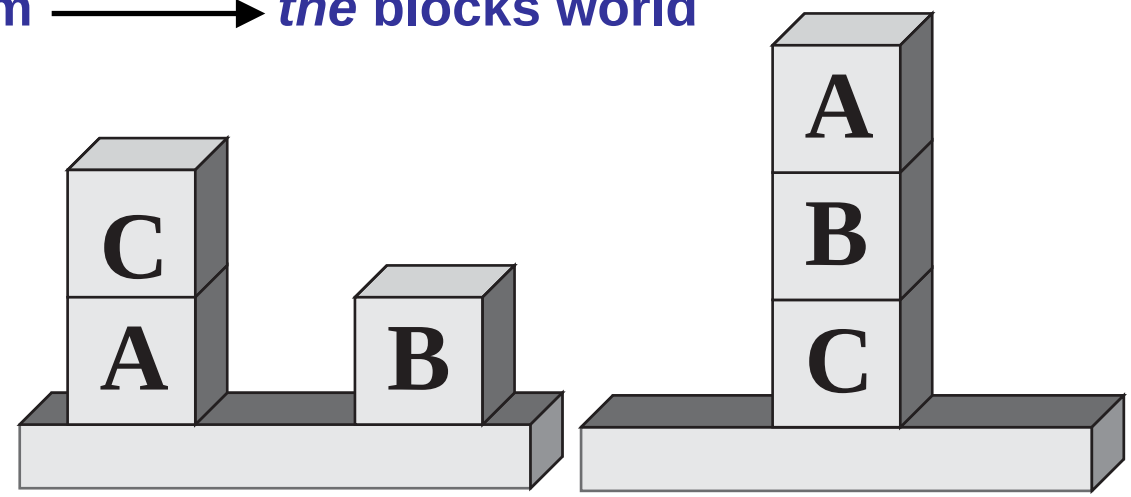
Good Old Fashioned AI: GOFAI

intelligence -- **look for essence**
study that
generalize back

attain goals

(and (on A B) (on B C))

the **program** → ***the*** **blocks world**

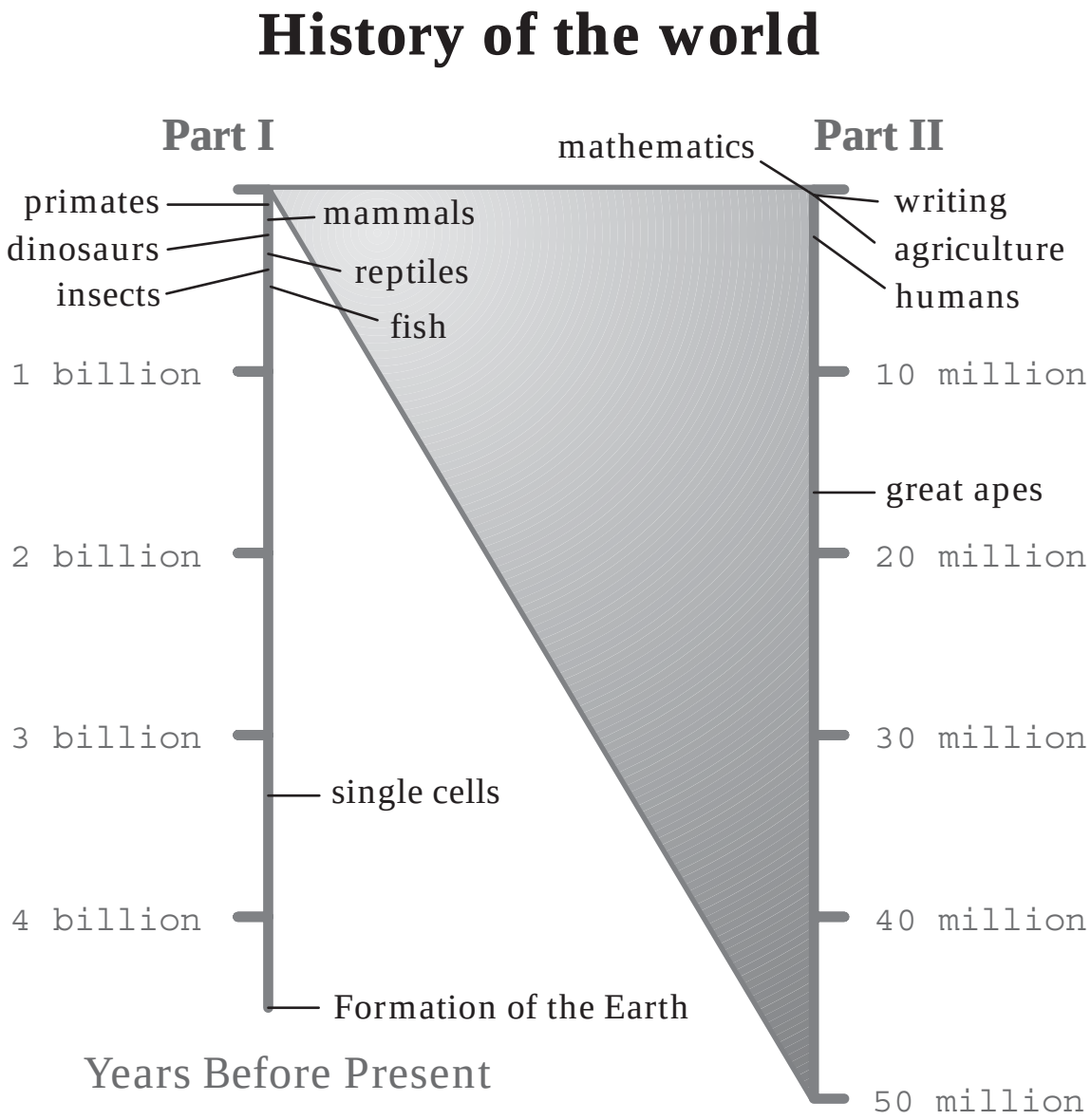


Marvin Minsky: *Society of Mind*

2.5 EASY THINGS ARE HARD

In attempting to make our robot work, we found that many everyday problems were much more complicated than the sorts of problems, puzzles, and games adults consider hard.

Where Did Evolution Spend Its Time?



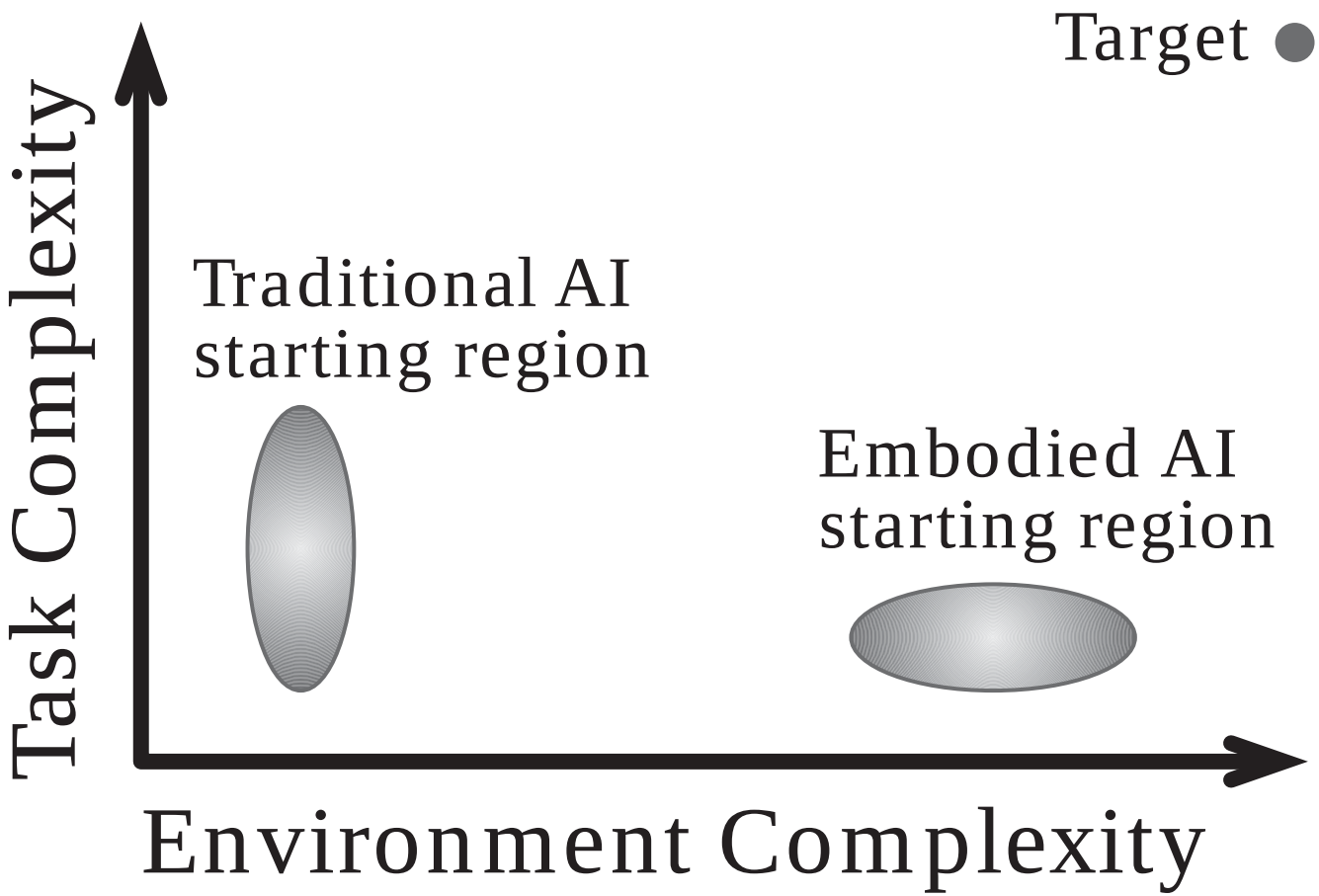
Creature, or Behavior-Based, AI

creatures -- **live in messy worlds**
performance relative to the world
intelligence (emerges) on this substrate

***the* creature** → ***all* possible worlds**

maintain goals

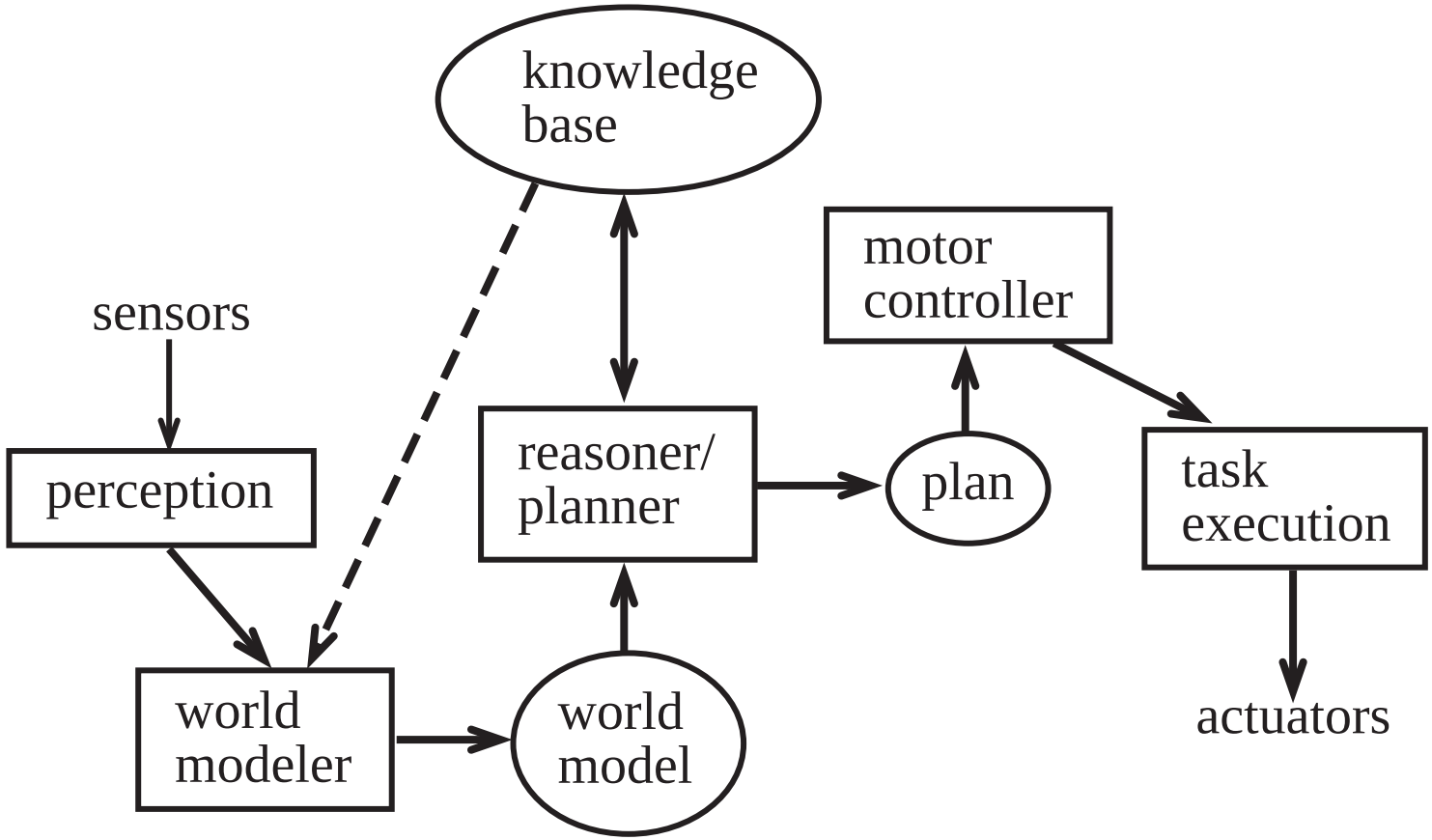
Methodologies Compared



Embrace *Hubris*

While it turns out that biological systems often use simple tricks to accomplish their goals, they are often more subtle than human engineers with all their mathematics and power tools may think they are.

Sense-Model-Plan-Act



Contrast: Thinking about Creatures

- Simple creatures occupy very complex worlds
  - they are not all knowing masters of the worlds
  - they act enough to capitalize on specific features of the world
- They do not have enough neurons to build full reconstructions of the world
- The `diameter' of their nervous systems is very small (about six for humans)

Herbert Simon's Ant

A man, viewed as a behaving system, is quite simple. The apparent complexity of his behavior over time is largely a reflection of the complexity of the environment in which he finds himself.

Embrace *Situatedness*

The behavior of a creature, depends on the environment in which it is embedded or situated.

Creatures don't deal with abstract descriptions, but with the "here" and "now" of their environment

Embrace *Embodiment*

An *embodied* creature is one which has a physical body and experiences the world, at least in part, directly through the influence of the world on that body.

The actions of a creature are part of a dynamic with the world and have immediate feedback on the creature's own sensations through direct physical coupling and its consequences.

Look for Emergence

The intelligence of the system emerges from the system's interactions with the world and from sometimes indirect interactions between its components--it is sometimes hard to point to one event or place within the system and say that is why some external action was manifested.

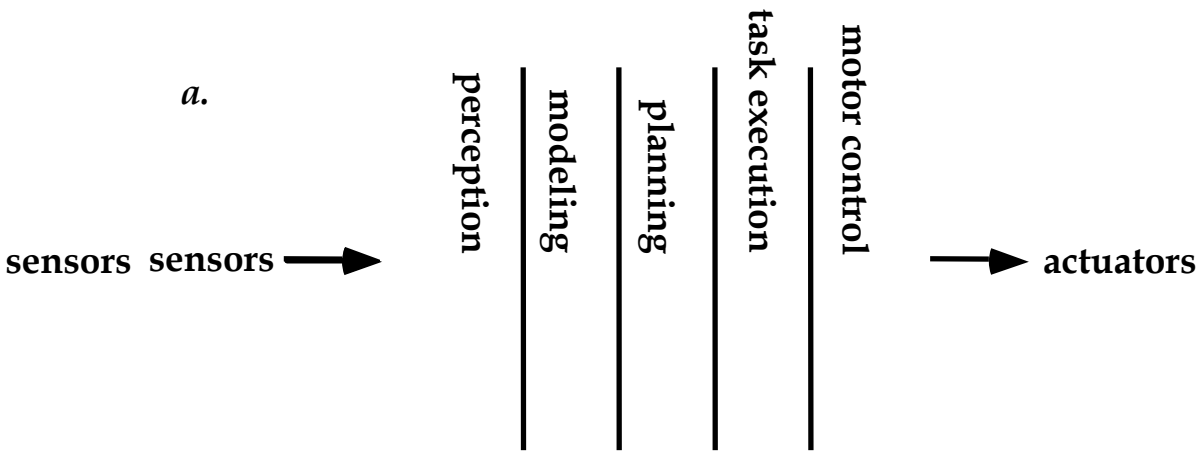
Autonomous

An *autonomous* (artificial) creature is one that is able to maintain a long term dynamic with its environment without intervention. Once an autonomous artificial creature is switched on, it does what is in its nature to do.

Distinguish the *Observer* from the *Robot*

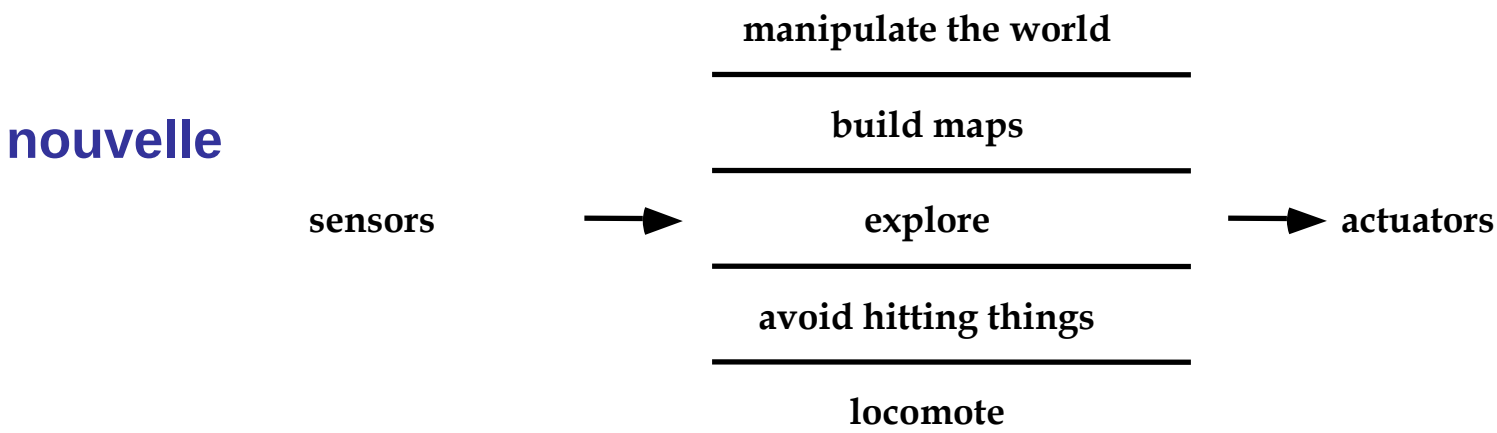
Terms descriptive of behavior are in the eye of the observer.

Traditional Problem Decomposition

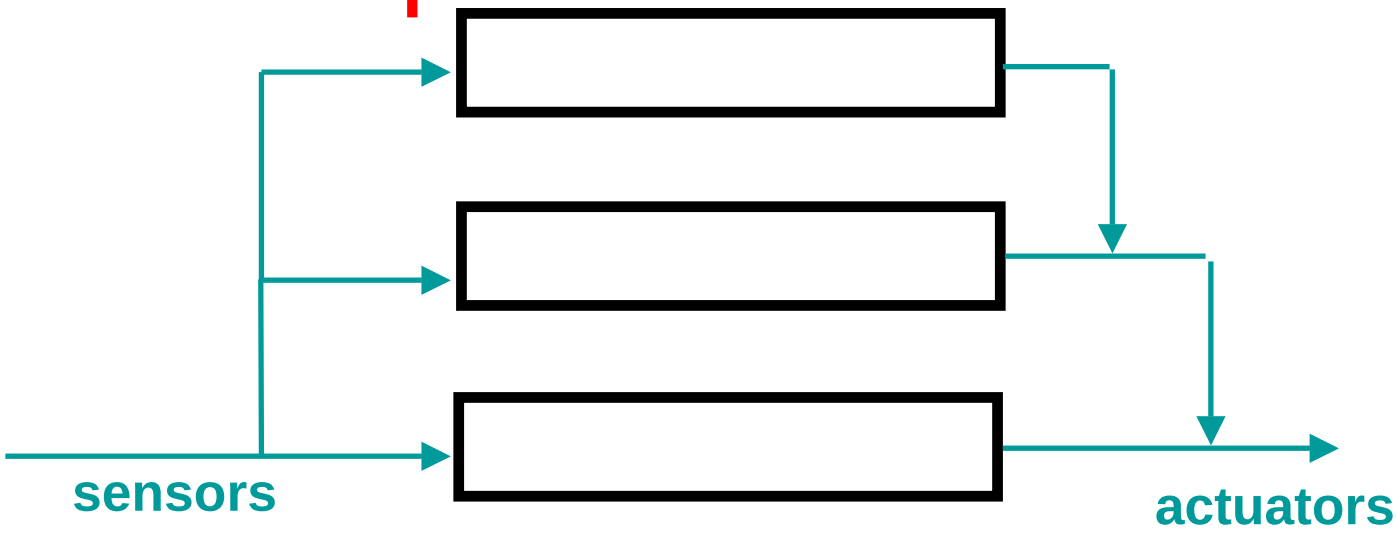


Horizontal decomposition

Behavior Based Decomposition



Recapitulate Evolution



- each layer has some perception, ‘planning’, and action
- rather than sensor fusion, we have sensor fission
- fusion happens at the action command level on the right
- there is a question of what sort of merge semantics there should be
- in its pure form, construction is purely additive

Suitable for Mobile Robots

- Handles multiple goals via different behaviors, with mediation, running concurrently
- Multiple sensors are not combined but complementary
- Robust: graceful degradation as upper layers are lost
- Additivity facilitates easy expansion for hardware resources

III. The Practicalities

- How should the task be decomposed?
  - Not a science!
- On behaviors and arbitration
- How should it be debugged?
- What will bite you!

Behavior Decomposition[1]

- State the problem clearly
- Identify any unstated assumptions about human competency that robot may not have
- State simply the set of minimum competencies needed to achieve the task
- Look for methods that will enable each competency using your robot h/w
- Match the questions that should be asked with sensors that can answer them
- Write behaviors that implement the methods and connect the behaviors to fixed priority arbiters
- Assume sensors will be noisy! Plan for graceful degradation
- Accept methods that, on average, advance the task
- Strive for robustness ahead of efficiency

1. From p 173, Jones, “Robot Programming: A Practical Guide to BB Robotics

On Behaviors

- Whenever (X) do
  - Else-whenever(Y) do
    - Etc
- Always sensing, looks for trigger then exerts control:
  - A behavior always monitors specific sensors,
  - it uses a threshold of their values to dictate when it will attempt to control a set of actuators: TRIGGER

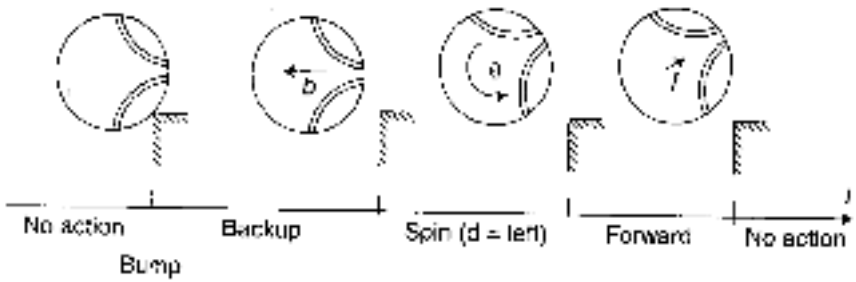
Servo vs Ballistic Behaviors

- Servo behavior has a feedback loop
  - Eg: light-positioning behavior
  - Never completes
- Ballistic behavior, once triggered continues to completion without any sensing
  - Eg: Escape behavior
    - 1. Back up a preset distance
    - 2. Spin a preset number of degrees
    - 3. Move forward a preset distance
  - Use with caution due to sequential nature
  - Try to solve with servo behavior first

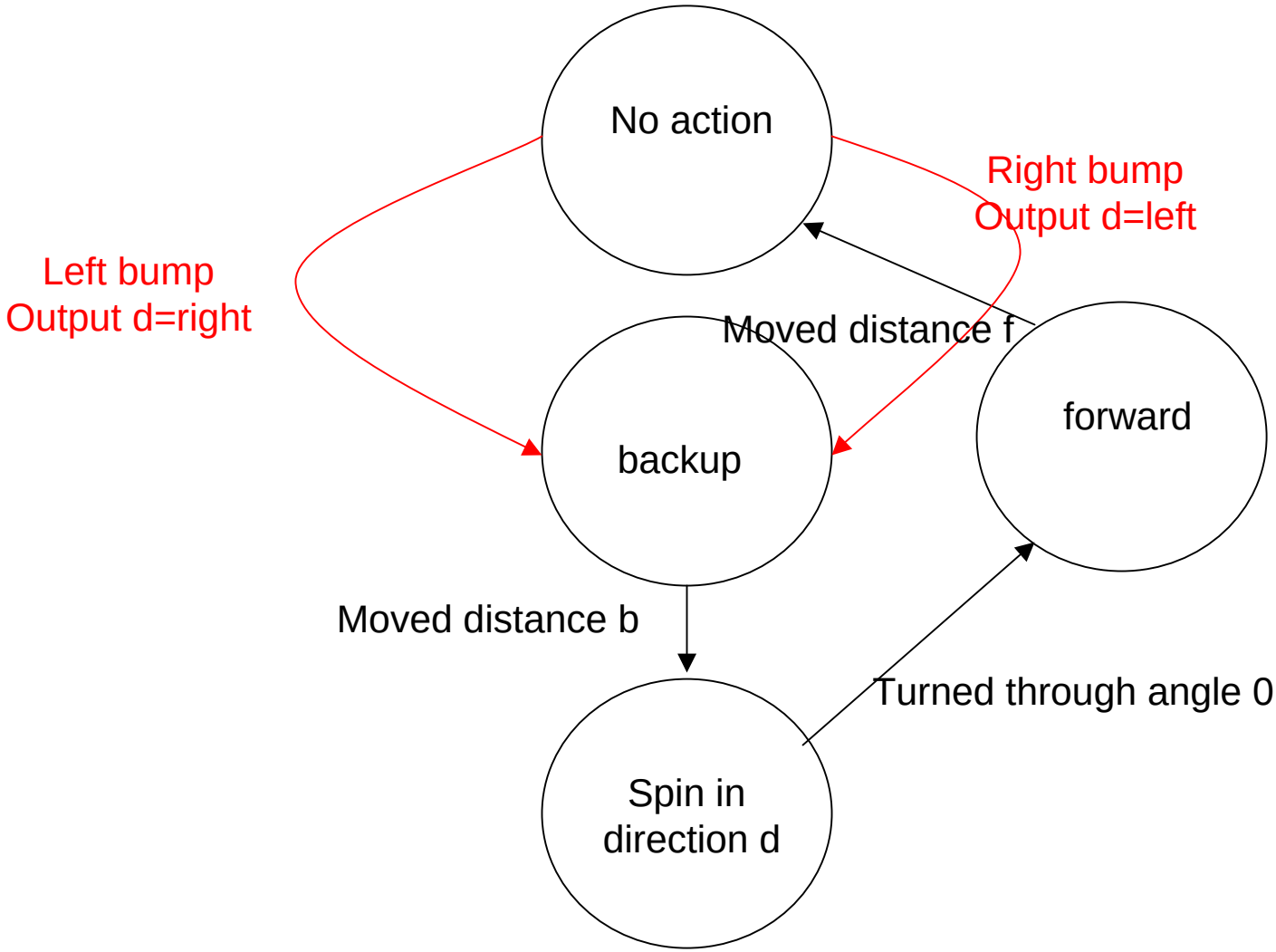
Using Finite State Machines for Design

- Behaviors have no (or little) state
  - They live in the ‘here’ and ‘now’ without memory
  - Use an FSM to for analysis and design to see how every event is being handled

Escape Diagrammed



Escape Behavior FSM

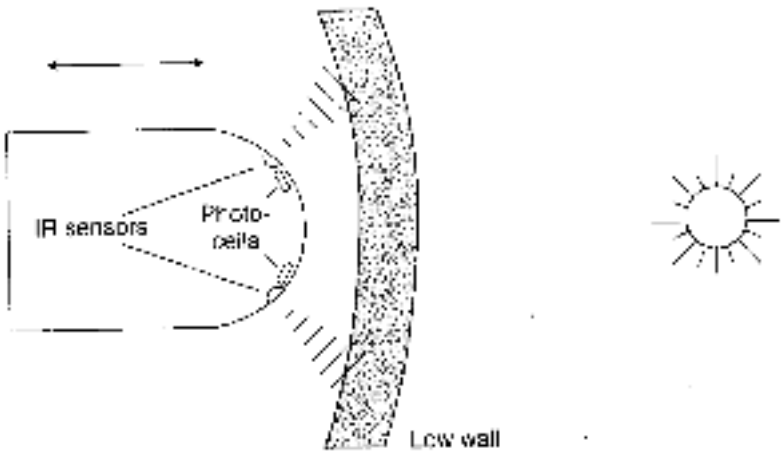


Overloading Behaviors

- What to do with behavior 1 is not distinct from behavior 2:
  - Eg. While reacting to one collision another occurs (while escape is running)
  - Don’t add in special cases “overloading a behavior”
  - Create a third behavior that looks for the trigger of behaviors 1 & 2, and controls that situation

Thrashing

- Two different behaviors are alternatively given control or two parts of one behavior contradict each other.



Thrashing Remedies

- Remedy: cycle-detection behavior
  - A series of rapid back and forth wheel motions or lack of progress
- Remedy: Table analysis,

On Arbitration

- When to arbitrate:
  - Eg. wander-behavior and recharge-behavior
    - What to decide? Average, take turns, vote
    - Use urgency
    - Consider graceful degradation
- Use fixed priority arbitration for most cases
- Can have multiple arbiters for different actuators
- Arbiter can report how it arbitrated

Debugging

- Develop and test each behavior in turn
- The difficulty will lie in understanding and managing the interactions between behaviors
- Example: thrashing
- Set up a debug tool: indicated which behavior is active, sensor values, state of arbiter
  - Could be tones or GUI

Wrap-Up

- Example
- Overview
- Practicalities
- Next
  - Consider implementation with Carmen and Java
  - Consider BB approach for challenge
  - More sophistication in BB creature - mapping
  - Subsumption: Example instance of BB design

Primary Source Material

- Brooks, R. A., *"New Approaches to Robotics"*, Science (253), September 1991, pp. 1227-1232.
- Brooks, R. A. and A. M. Flynn *"Fast, Cheap and Out of Control: A Robot Invasion of the Solar System"*, Journal of the British Interplanetary Society, October 1989, pp. 478≦485.
- Brooks, R. A. *"A Robust Layered Control System for a Mobile Robot"*, IEEE Journal of Robotics and Automation, Vol. 2, No. 1, March 1986, pp. 14-23; also MIT AI Memo 864, September 1985.
- Robot Programming: A Practical Guide to Behavior-based Robotics, Joseph L. Jones, McGraw-Hill, 2004.
- Lecture #1, Introduction, Prof. Ian Horswill http://www.cs.northwestern.edu/academics/courses/special_topics/395-robotics/
- “Sensing and Manipulating Built-for-Human Environments”, Brooks et al, International Journal of Humanoid Robotics, Vol 1, #1, 2004.