AOT LAB

Agent and **O**bject **T**echnology **Lab**
Dipartimento di Ingegneria dell'Informazione
Università degli Studi di Parma

Multi-Agent Systems

Agent Architectures

Prof. Agostino Poggi

Ipothesis

- Assume the environment may be in any of a finite set E of discrete, instantaneous states

$$E = \{e, e', \ldots\}$$

- Agents are assumed to have a repertoire of possible actions available to them, which transform the state of the environment

$$Ac = \{\alpha, \alpha', \ldots\}$$

- Therefore, the live (run), r, of an agent in an environment can be described by a sequence of interleaved environment states and actions

$$r : e_0 \xrightarrow{\alpha_0} e_1 \xrightarrow{\alpha_1} \ldots \xrightarrow{\alpha_{n-1}} e_n$$

Purely Reactive Agents

- Some agents decide what to do without reference to their history
  - They base their decision making entirely on the present
  - With no reference at all to the past
- We call such agents purely reactive

$$action : E \rightarrow Ac$$

- A thermostat is a purely reactive agent

$$action(e) = off \quad \text{if } e = \text{temperature OK}$$
$$action(e) = on \quad \text{otherwise}$$

Purely Reactive Agents

- The see function is the agent’s ability to observe its environment
- Output of the see function is a percept

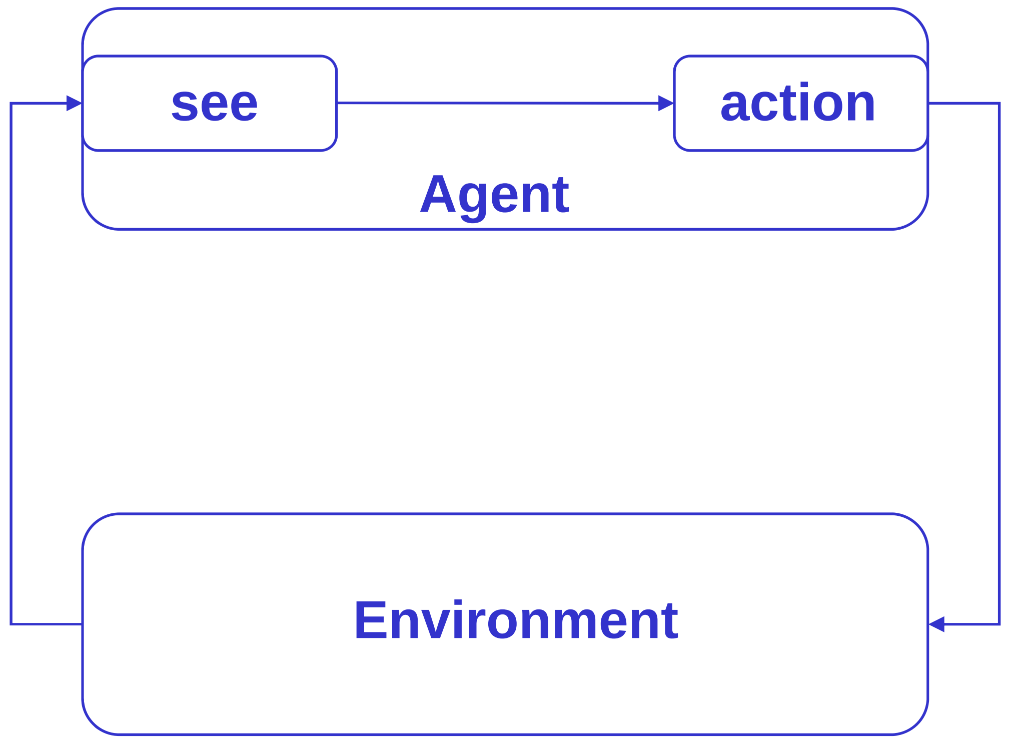
  $see : E \rightarrow P$

  which maps environment states to a set of percepts
- The action function represents the agent’s decision making process

  $action : P \rightarrow Ac$

  which maps percepts to actions

Purely Reactive Agents



AOT
LAB

Agents with State

- Agents with state maintains information about the environment state and history
- Let *S* be the set of all internal states of the agent
- The perception function see for a state based agent is unchanged

  $$see : E \rightarrow P$$

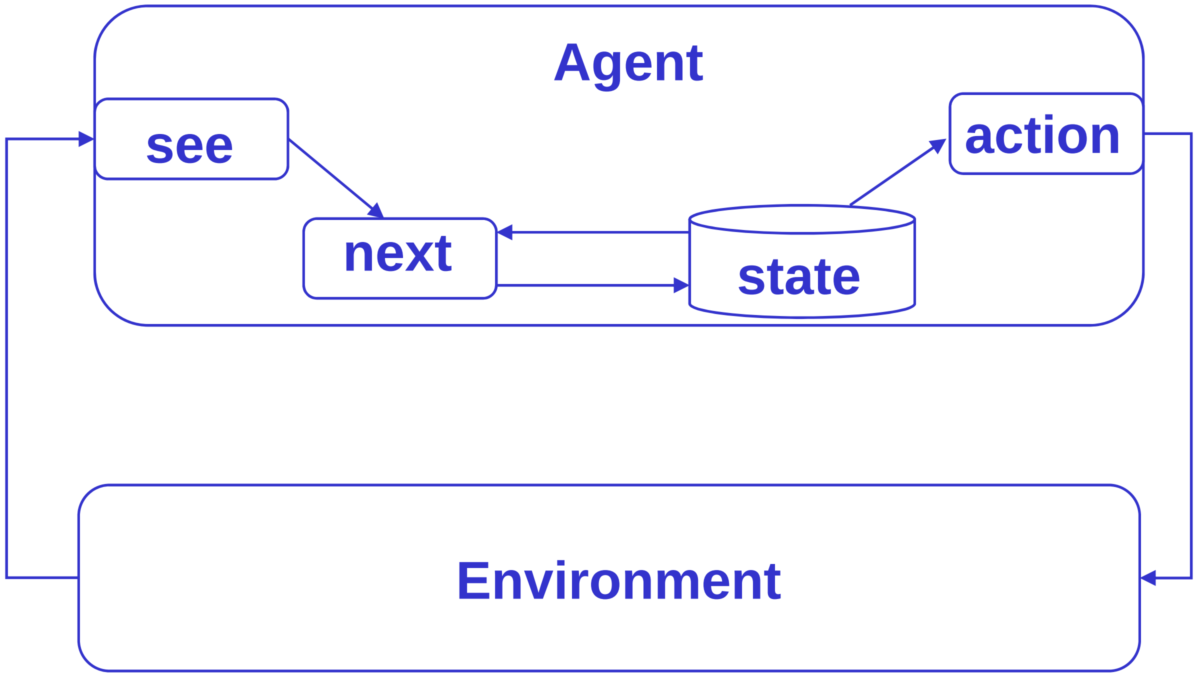
- The action selection function action is now defined as a mapping from internal states to actions

  $$\text{action} : S \rightarrow Ac$$

- A new function *next* maps internal states and percepts to internal states

  $$next : S \times P \rightarrow S$$

Agents with State



Agent Control loop

1. Agent starts in some initial internal state s_0
2. Observes its environment state *e* and generates a percept: *see*(*e*)
3. Updates its internal state: next(s_0 , *see*(*e*))
4. Selects the action: *action*(*next*(s_0 , *see*(*e*)))
5. Executes the selected action
6. Goto step (2)

Types of Architectures

- We want to build agents, that enjoy the properties of autonomy, reactivity, pro-activeness, and social ability
- To do it, we can use four main types of agent architectures
  - Deliberative
  - BDI
  - Reactive
  - Hybrid

Deliberative Agents

- The classical approach to building agents is
  - To view agents as a particular type of knowledge-based system
  - Bring all the associated methodologies of such systems
- This paradigm is known as symbolic AI
- We define a deliberative agent or agent architecture to be one that
  - Contains an explicitly represented, symbolic model of the world
  - Makes decisions (for example about what actions to perform) via symbolic reasoning

Deliberative Agents Realization Problems

- The transduction problem
  - Translating the real world into an accurate, adequate symbolic description, in time for that description to be useful (vision, speech understanding, learning , ...)
- The representation/reasoning problem
  - How to symbolically represent information about complex real-world entities and processes
  - How to get agents to reason with this information in time for the results to be useful (knowledge representation, automated reasoning, automatic planning, ...)

AOT LAB

Action Selection

- How can an agent decide what to do using theorem proving?
- Basic idea is to use logic to encode a theory stating the best action to perform in any given situation
- Let
  - ρ be this theory (typically a set of rules)
  - Δ be a logical database that describes the current state of the world
  - *Ac* be the set of actions the agent can perform
  - $\Delta \vdash_{\rho} \Phi$ mean that Φ can be proved from Δ using ρ

Action Selection

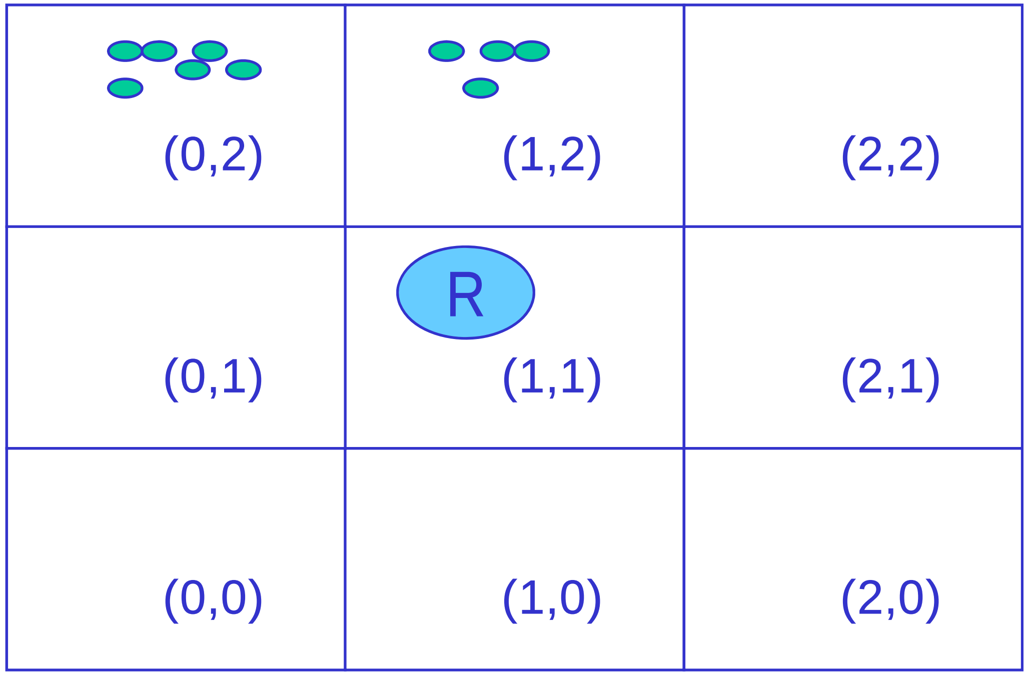
```
for each a $\in$ Ac do
    if $\Delta \vdash_{\rho} Do(a)$ then
      return a
    end-if
end-for

for each a $\in$ Ac do
    if $\Delta \vdash_{\rho} \neg Do(a)$ then
      return a
    end-if
end-for

return null
```

try to find an action explicitly prescribed

try to find a no excluded action

AOT LAB

The Vacuum World



The Vacuum World

- Goal is for the robot to clear up all dirt
- Domain is represented by the following predicates:

In (x, y)	agent is at (x, y)
Dirt(x, y)	there is dirt at (x, y)
Facing(d)	the agent is facing direction d

- Possible actions are:

  Ac = {turn, forward, suck}

  where turn means “turn right”

The Vacuum World

- Rules for determining what to do:

  $In(0,0) \wedge Facing(north) \wedge \neg Dirt(0,0) \rightarrow Do(forward)$

  $In(0,1) \wedge Facing(north) \wedge \neg Dirt(0,1) \rightarrow Do(forward)$

  $In(0,2) \wedge Facing(north) \wedge \neg Dirt(0,2) \rightarrow Do(turn)$

  $In(0,2) \wedge Facing(east) \rightarrow Do(forward)$

  . . . and so on!

- Using these rules (together other obvious ones), starting at (0, 0) the robot will clear up dirt

AOT
LAB

Pros & Cons

- Simple
- Elegant
- Logical semantics
- Representation and deliberation complexity
- Not applicable in (fast) changing environments

How to Enhance Logical Agents?

- Weaken the logic
- Use symbolic, non-logical representations
- Shift the emphasis of reasoning from run time to design time

AOT
LAB

BDI Agents

- BDI architecture aims to model rational or intentional agents
- A BDI architecture addresses how beliefs, desires and intentions are represented, updated and executed through a practical reasoning process
- In a BDI agent intensions are represented by plans that can be defined composing actions and/or predefined pieces of plan

Practical Reasoning

- Practical reasoning is reasoning directed towards actions

- Bratmam said that practical reasoning is a matter of weighing conflicting considerations for and against competing options, where the relevant considerations are provided by
  - What the agent desires/values/cares about
  - What the agent believes

- Practical reasoning differs from theoretical reasoning because it is directed towards beliefs

Practical Reasoning

- Human practical reasoning consists of two activities
  - Deliberation
    - Deciding what state of affairs we want to achieve
  - Means-ends reasoning
    - Deciding how to achieve these states of affairs
- The outputs of deliberation are intentions

Intensions

- Intentions pose problems for agents, who need to determine the ways of achieving them
  - If I have an intention, you would expect me to devote resources to deciding how to bring about this intension

- Intentions provide a “filter” for adopting other intentions, which must not conflict
  - If I have an intention, you would not expect me to adopt another intention that is mutually exclusive with the other intensions
  - No conflicting intentions should be adopted!

AOT
LAB

Intensions

- Agents track the success of their intentions, and are inclined to try again if their attempts fail
  - If an agent’s first attempt to achieve a goal fails, then it will try an alternative plan to achieve the goal, if all other things will remain equal
- Agents believe their intentions are possible
  - That is, they believe there is at least some way that the intentions could be brought about

Intentions

- Agents do not believe they will not realize their intentions
  - It would not be rational of me to adopt an intention if I believed this intension was not possible
- Agents believe they will bring about their intentions under certain circumstances
  - It would not normally be rational of me to believe that I would bring my intentions about, but intentions can fail
  - Moreover, it does not make sense that if I believe that an intention is realizable, then is inevitable that I would adopt it as an intention

AOT LAB

Intensions

- Agents need not intend all the expected side effects of their intentions
  - If I believe $\Phi \Rightarrow \psi$ and I intend that Φ, I do not necessarily intend also ψ (Intentions are not closed under implication)
  - This last problem is known as the side effect or package deal problem
    - I may believe that going to the dentist involves pain
    - I may also intend to go to the dentist
    - But this does not imply that I intend to suffer pain!

AOT
LAB

Intensions Are Stronger Than Desires

- My desire to play basketball this afternoon is merely a potential influencer of my conduct this afternoon
- It must vie with my other relevant desires [. . .] before it is settled what I will do
- In contrast, once I intend to play basketball this afternoon, the matter is settled: I normally need not continue to weigh the pros and cons
- When the afternoon arrives, I will normally just proceed to execute my intentions (Bratman, 1990)

AOT
LAB

BDI Agent Control Loop I & II

```
$B = B_0$;
$I = I_0$;

while  true  do
  observe the world;
  update internal state;
  select an intention;
  get a plan for the intention;
  execute the plan;
end while
```

```
$B = B_0$;
$I = I_0$;

while  true  do
  get next percept p;
  $B = brf(B, p)$;
  $I = deliberate(B)$;
  $\pi = plan(B, I)$;
  $execute(\pi)$;
end while
```

Time Constraints

- Problem: deliberation and means-ends reasoning processes are not instantaneous
  - They have a *time cost*
- Suppose the agent
  - Starts deliberating at t_0
  - Starts means-ends reasoning at t_1
  - Starts executing the plan at time t_2
- Time to deliberate is

$$t_{deliberative} = t_1 - t_0$$

- Time for means-ends reasoning is

$$t_{mer} = t_2 - t_1$$

Time Constraints

- Further suppose that deliberation is optimal in that if it selects some intention to achieve, then this is the best thing for the agent (maximises expected utility)
- So at time t_1, the agent has selected an intention to achieve that would have been optimal if it had been achieved at t_0
- But unless $t_{deliberative}$ is vanishingly small, then the agent runs the risk that the intention selected is no longer optimal by the time the agent has fixed upon it
- Deliberation is only half of the problem
  - The agent still has to determine how to achieve the intention

Optimal Behavior

- When deliberation and means-ends reasoning take a vanishingly small amount of time
- When the world is guaranteed to remain static while the agent is deliberating and performing means-ends reasoning
  - Assumptions upon which the choice of intention to achieve and plan to achieve the intention remain valid until the agent has completed deliberation and means-ends reasoning
- When an intention that is optimal when achieved at the time at which the world is observed (t_0) is guaranteed to remain optimal until the time (t_2) at which the agent has found a course of action to achieve the intention

Deliberation

- How does an agent deliberate?
  - Begin by trying to understand what the *options* available to the agent are
  - *Choose between them*, and *commit* to some
- Chosen options are then intentions

Deliberation Components

- *Option generation,* in which the agent generates a set of possible alternatives (desires) that can be achieved
  - Represent option generation via a function, *options*, which takes the agent's current beliefs and current intentions, and from them determines a set of options
- *Filtering,* in which the agent chooses between competing alternatives and commits to achieving them
  - In order to select between competing options, an agent uses a *filter* function

AOT
LAB

BDI Agent Control Loop III

```
$B = B_0$;
$I = I_0$;
while  true  do
    get next percept p;
    B = brf(B, p);
    I = deliberate(B);
    π = plan(B, I);
    execute(π);
end while
```

```
D = options(B, I);
I = filter(B, D, I);
```

Plan Failure

- An agent has commitment both to
  - *Ends* (i.e., the state of affairs it wishes to bring about)
  - M*eans* (i.e., the mechanism via which the agent wishes to achieve the state of affairs)
- Currently, our agent control loop is overcommitted, both to means and ends
- Modification
  - *Replan* if ever a plan goes wrong

AOT
LAB

BDI Agent Control Loop IV

```
B = $B_0$;
I = $I_0$;
while  true  do
    get next percept p;
    B = brf(B, p);
    D = options(B, I);
    I = filter(B, D, I);
    $\pi$ = plan(B, I);
    execute($\pi$);
end while
```

```
while  true  do
    $\alpha$ = head($\pi$);
    execute($\alpha$);
    $\pi$ = tail($\pi$);
    get next percept p;
    B = brf(B,p);
    if not sound($\pi$, I, B) then
        $\pi$ = plan(B, I);
    end if
end while
```

Intension Success or Failure

- Still overcommitted to intentions
  - Never stops to consider whether or not its intentions are appropriate
- Modification
  - Stop to determine whether intentions have succeeded or whether they are impossible
    - Single-minded commitment

AOT LAB

Commitment Strategies

- ***Blind commitment**, a* blindly committed agent will continue to maintain an intention until it believes the intention has actually been achieved
  - Blind commitment is also sometimes referred to as *fanatical* commitment
- ***Single-minded commitment**, a* single-minded agent will continue to maintain an intention until it believes that either the intention has been achieved, or else that it is no longer possible to achieve the intention
- ***Open-minded commitment**, an* open-minded agent will maintain an intention until it is its desire

AOT
LAB

BDI Agent Control Loop V

```
B = $B_0$;
I = $I_0$;
while  true  do
    get next percept p;
    B = brf(B, p);
    D = options(B, I);
    I = filter(B, D, I);
    π = plan(B, I);
    …
 end while
```

```
while  true  do
    α = head(π);
    execute(α);
    π = tail(π);
    get next percept p;
    B = brf(B,p);
    if not sound(π, I, B) then
        π = plan(B, I);
    end if
end while
```

```
while not empty(π)
   or succeeded(B, I)
   or impossible(B, I)
do
```

Intension Reconsideration

- Our agent gets to reconsider its intentions once every time around the outer control loop, i.e., when
  - It has completely executed a plan to achieve its current intentions
  - It believes it has achieved its current intentions
  - It believes its current intentions are no longer possible
- This is limited in the way that it permits an agent to *reconsider* its intentions
- Modification
  - Reconsider intentions after executing every action

AOT LAB

BDI Agent Control Loop VI

```
B = $B_0$;
I = $I_0$;
while  true  do
    get next percept p;
    B = brf(B, p);
    D = options(B, I);
    I = filter(B, D, I);
    π = plan(B, I);
    ...
end while
```

```
while not empty(π)
      or succeeded(B, I)
      or  impossible(B, I) do
  α = head(π);
  execute(α);
  π = tail(π);
  get next percept p;
  B = brf(B,p);
  ...
  if not sound(π, I, B) then
      π = plan(B, I);
  end if
end while
```

```
D = options(B, I);
I = filter(B, D, I);
```

Intension Reconsideration Cost

- But intention reconsideration is *costly*!
- A dilemma
  - An agent that does not stop to reconsider its intentions sufficiently often will continue attempting to achieve its intentions even after it is clear that they cannot be achieved, or that there is no longer any reason for achieving them
  - An agent that *constantly* reconsiders its intentions may spend insufficient time actually working to achieve them, and hence runs the risk of never actually achieving them
- Solution
  - Incorporate an explicit *meta-level control* component, that decides whether or not to reconsider

AOT LAB

BDI Agent Control Loop VII

```
B = $B_0$;
I = $I_0$;
while  true  do
    get next percept p;
    B = brf(B, p);
    D = options(B, I);
    I = filter(B, D, I);
    π = plan(B, I);
    …
end while
```

```
while not empty(π)
    or succeeded(B, I)
    or  impossible(B, I) do
  α = head(π);
  execute(α);
  π = tail(π);
  get next percept p;
  B = brf(B,p);
  …
  if not sound(π, I, B) then
     π = plan(B, I);
  end if
end while
```

```
if reconsider(I, B)
    then
    D = options(B, I);
    I = filter(B, D, I);
end if
```

```
D = options(B, I);
I = filter(B, D, I);
```

AOT LAB

Optimal Intension Reconsideration

- Kinny and Georgeff's experimentally investigated effectiveness of intention reconsideration strategies

- Two different types of reconsideration strategy were used
  - *Bold* agents, never pause to reconsider intentions
  - *Cautious* agents, stop to reconsider after every action

AOT
LAB

Optimal Intension Reconsideration

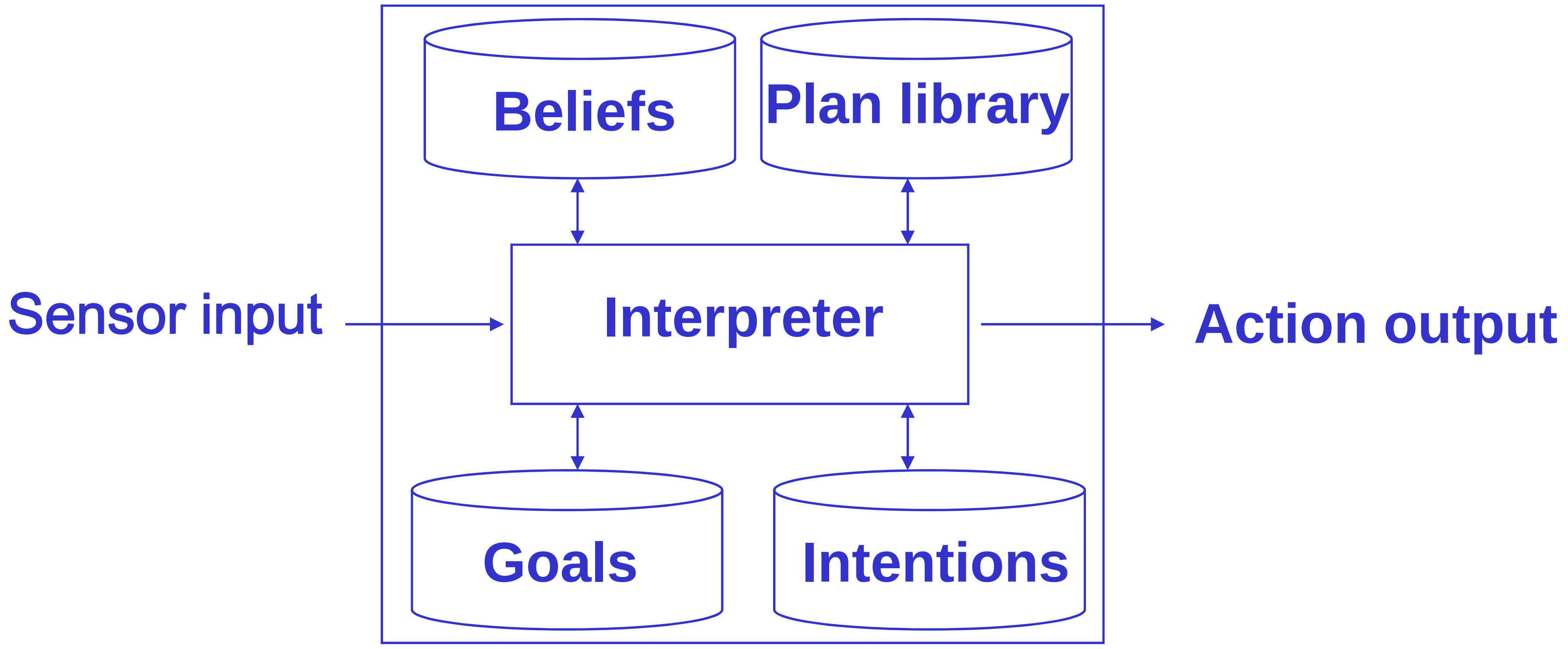
- *Let represent* dynamism in the environment with the rate of world change, γ
- If γ is low (i.e., the environment does not change quickly), then bold agents do well compared to cautious ones
  - Cautious agents waste time reconsidering their commitments
  - Bold agents are busy working towards — and achieving — their intentions
- If γ is high (i.e., the environment changes frequently), then cautious agents tend to outperform bold agents
  - Cautious agents are able to recognize when intentions are doomed, and also to take advantage of serendipitous situations and new opportunities when they arise

AOT
LAB

BDI Implementations

- There are different implementation of the BDI architecture
  - Procedural Reasoning System (PRS)
  - Intelligent Resource-bounded Machine Architecture (IRMA)
  - dMars
  - Jack
  - Jadex

PRS


Beliefs
Plan library
Sensor input
Interpreter
Action output
Goals
Intentions

Plan Library

- In the PRS, each agent is equipped with a *plan library*, representing agent's *procedural knowledge*
  - Knowledge about the mechanisms that can be used by the agent in order to realize its intentions
- The options available to an agent are directly determined by the plans an agent has
  - An agent with no plans has no options

Plan Components

- Invocation condition: circumstances for plan consideration
- Context: circumstances for successful plan execution
- Maintenance condition: must be true while plan is executing, in order for it to succeed
- Body: course of action, consisting is a tree of states and branches where branches represent goals or actions

Pros & Cons

- Architecture behavior is intuitive
- Clear functional decomposition
- Low efficiency
- It is difficult to balance commitment with reconsideration

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

Behavior Languages

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

Behavior Languages

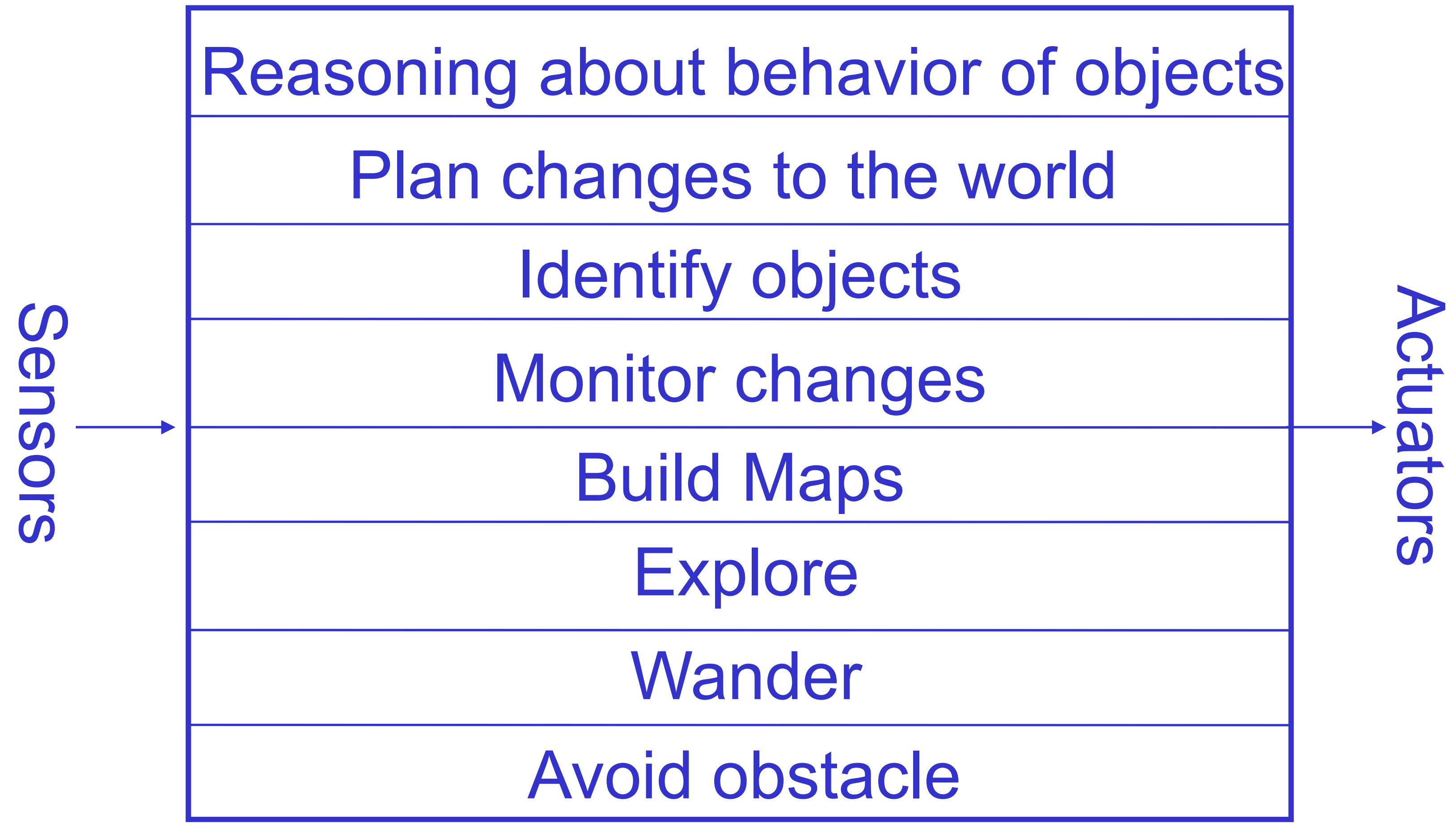
- He identifies two key ideas that have informed his research
  - Situatedness and embodiment: ‘Real’ intelligence is situated in the world, not in disembodied systems such as theorem provers or expert systems
  - Intelligence and emergence: ‘Intelligent’ behavior arises as a result of an agent’s interaction 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*

AOT LAB

Subsumption Architecture

- A subsumption architecture is a hierarchy of task-accomplishing *behaviors*
- Each behavior is a rather simple rule-like structure
- Each behavior ‘competes’ with others to exercise control over the agent
- Lower layers represent more primitive of (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

Subsumption Architecture



Steels’ Mars Explorer System

- 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*

AOT
LAB

Steels' Mars Explorer System

- 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
- Any samples carried by agents are dropped back at the mother-ship
  - *if* carrying samples *and* at the base *then* drop samples
- Agents carrying samples will return to the mother-ship
  - *if* carrying samples and *not* at the base *then* travel up gradient
- Agents will collect samples they find
  - *if* detect a sample *then* pick sample up
- An agent with "nothing better to do" will explore randomly
  - *if* true *then* move randomly

Pros & Cons

- Simplicity
- Economy
- Computational tractability
- Robustness against failure
- Elegance

Pros & Cons

- Agents without environment models must have sufficient information available from local environment
- If decisions are based on local environment, how does it take into account non-local information (i.e., it has a “short-term” view)
- Difficult to make reactive agents that learn
- Since behavior emerges from component interactions plus environment, it is hard to see how to engineer specific agents (no principled methodology exists)
- It is hard to engineer agents with large numbers of behaviors (dynamics of interactions become too complex to understand)

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
  - A *reactive* one, which is capable of reacting to events without complex reasoning

Hybrid Architectures

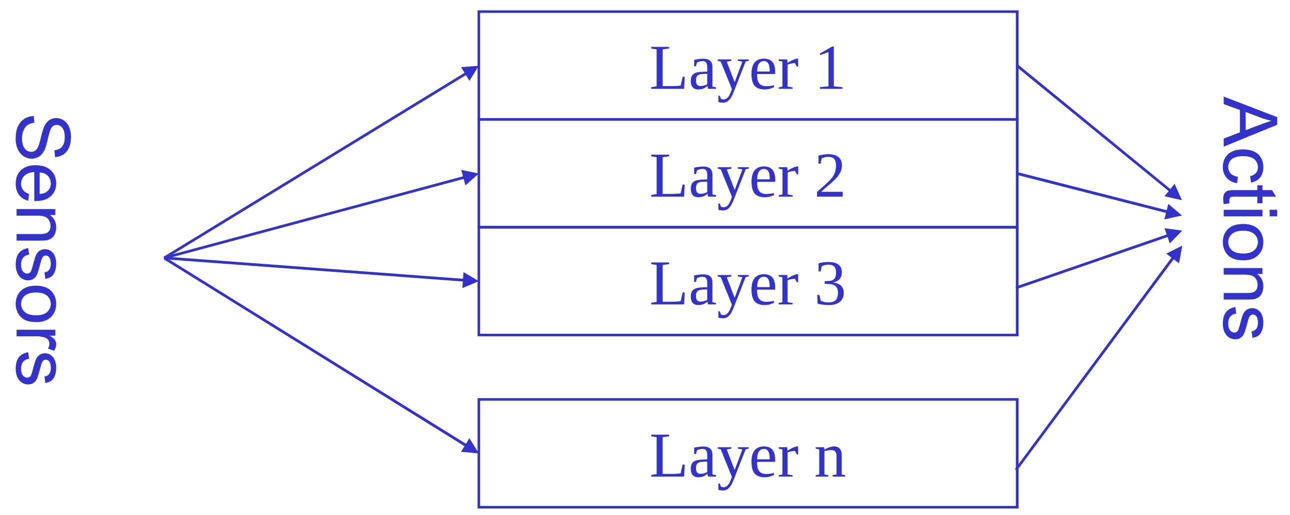
- Often, the reactive component is given some kind of precedence over the deliberative one
- This kind of structuring leads naturally to the idea of a *layered* architecture, of which TOURINGMACHINES and INTERRAP are examples
- In such an architecture, an agent’s control subsystems are arranged into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

AOT
LAB

Hybrid Architectures

- 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

Horizontal Layering

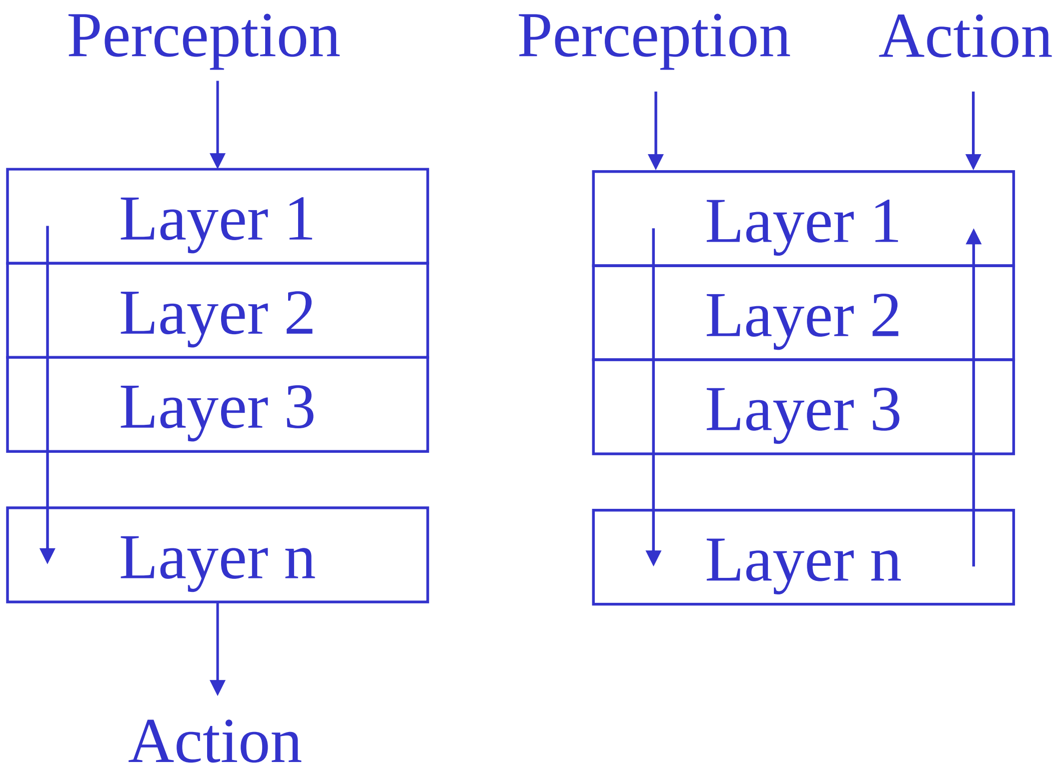


Horizontal Layering

- Great advantage is their conceptual simplicity
  - If we need an agent to exhibit n different types of behavior, then we implement n different layers!
- But the layers are in effect competing with one another
- Generally there is a mediator function making decisions about which layer has control of the agent at a given time
  - Point of failure of the system
  - Complex to be realized
    - All possible interactions between layers must be considered!

AOT LAB

Vertical Layering

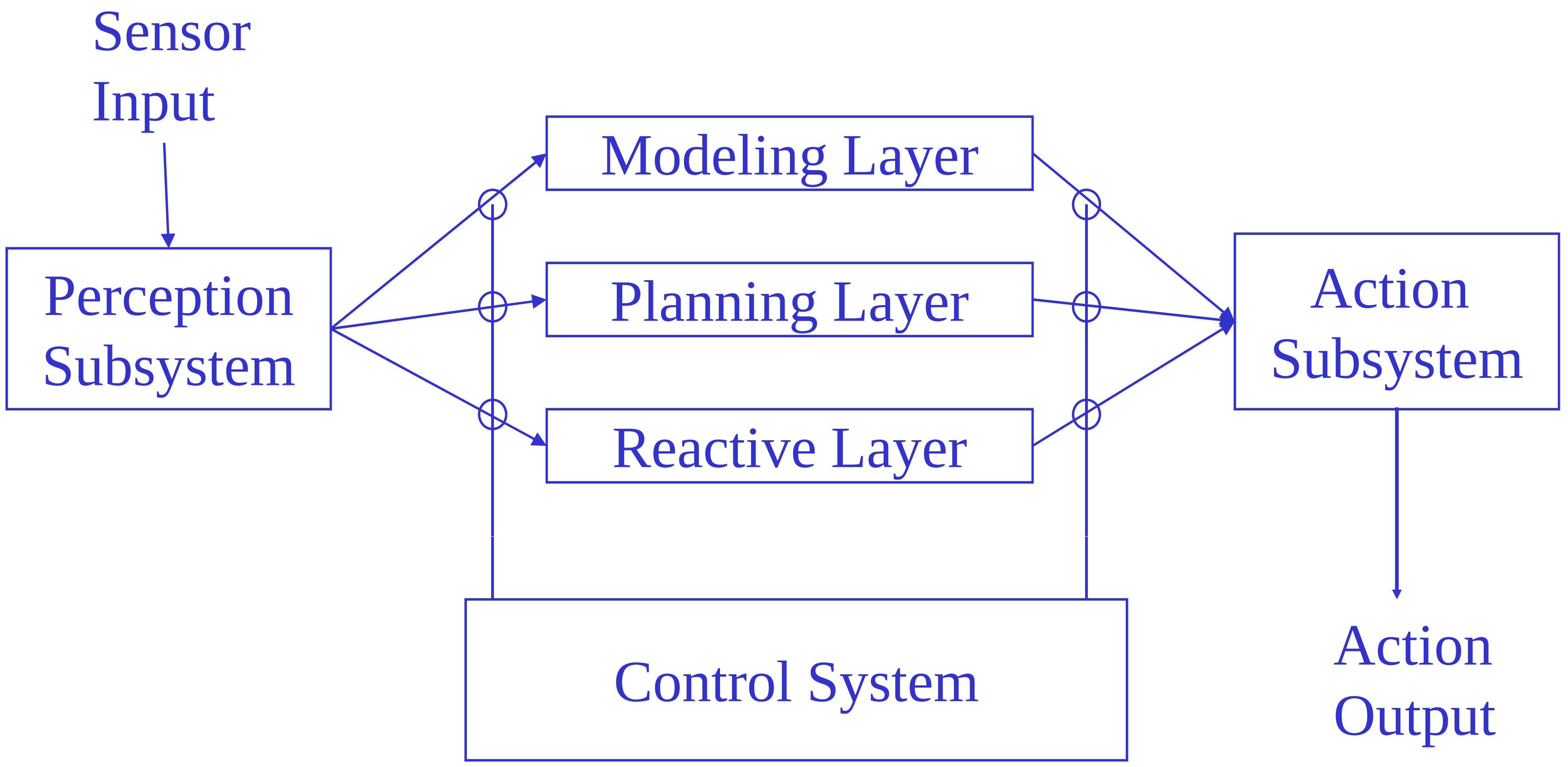


Vertical Layering

- Complexity of interactions between layers is reduced
  - If each layer is capable of suggesting m actions there are at most m^2 (n-1) interactions
  - In the same conditions, in the horizontal case there are m^n interactions
- But this architecture is less flexibility
  - Control must pass between each different layer
  - Less fault tolerant
    - Failures in any one layer are likely to have serious consequences for agent performance

AOT LAB

Ferguson's TOURINGMACHINES



Ferguson's TOURINGMACHINES

- The *reactive layer* is implemented as a set of situation-action rules, "*a la*" subsumption architecture
- The *planning layer* constructs plans and selects actions to execute in order to achieve the agent's goals
- The *modeling 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*

AOT LAB

Ferguson’s TOURINGMACHINES

- Example of reactive layer rule

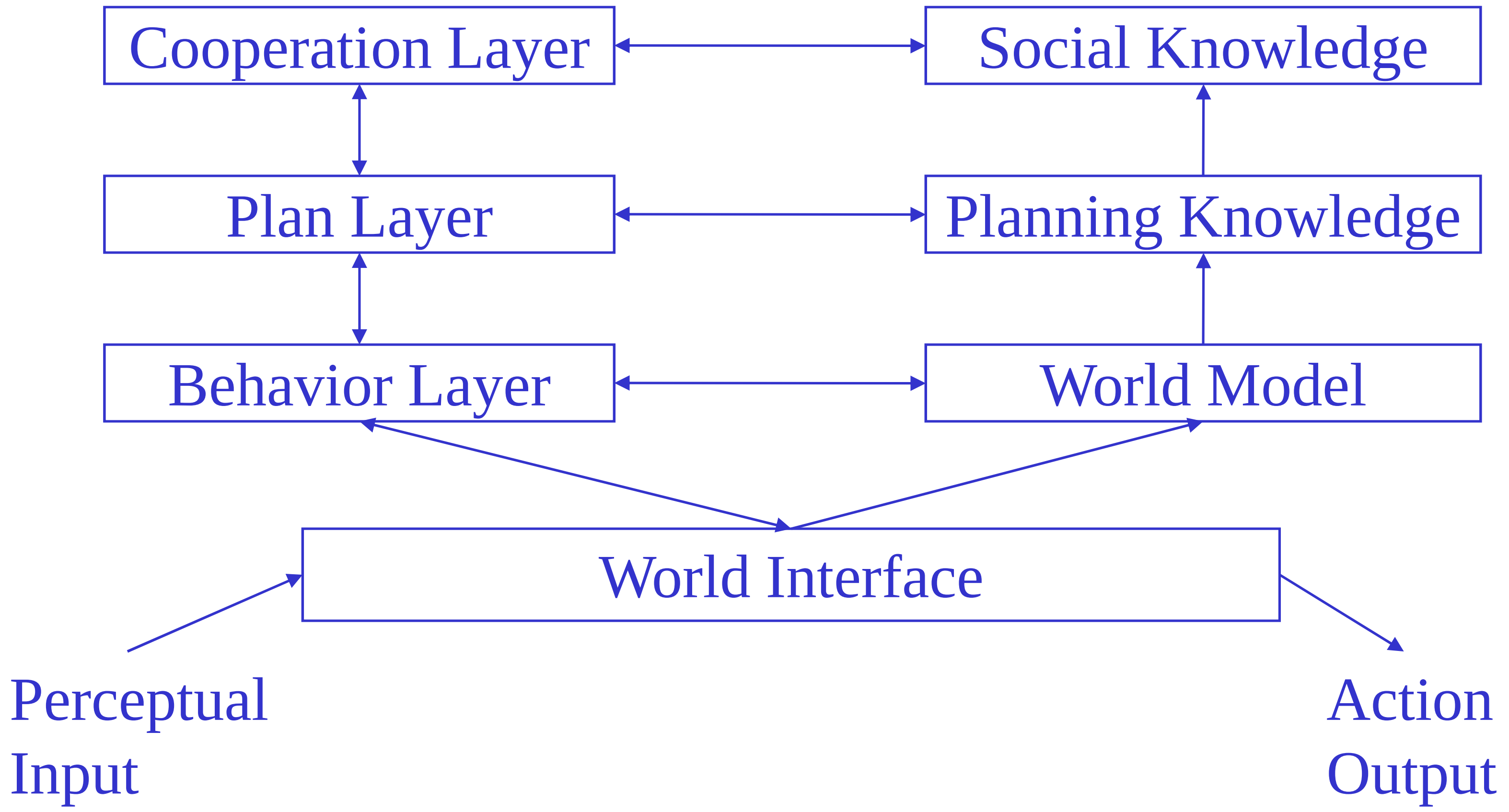
```
if
    is-in-front(Kerb, Observer) and
    speed(Observer) > 0 and
    separation(Kerb, Observer) < KerbThreshHold
then
    change-orientation(KerbAvoidanceAngle)
```

- Example of control subsystem rule

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

AOT
LAB

Müller’s InteRRaP



AOT
LAB

Müller's InteRRaP

- The purpose of the three layers appears to be rather similar to TouringMachine's layers
  - The behavior based layer deals with reactive behavior
  - Local planning deals with everyday planning
  - Co-operative planning deals with social interactions
- Bottom-up activation occurs when a layer passes control to a higher layer because it is not competent to deal with the current situation
- Top-down execution occurs when a higher layer makes use of the facilities provided by a lower layer to achieve one of its goals

Müller’s InteRRaP

1. The flow of control begins when the perceptual input arrives at the lowest layer

2. If the reactive layer can deal with this input
   - Then it manages the input
   - Otherwise, bottom-up activation occurs

3. Control is passed to the local planning layer

4. If this layer can handle the situation
   - Then it makes use of top-down execution
   - Otherwise, it uses bottom-up activation to pass control to the highest layer