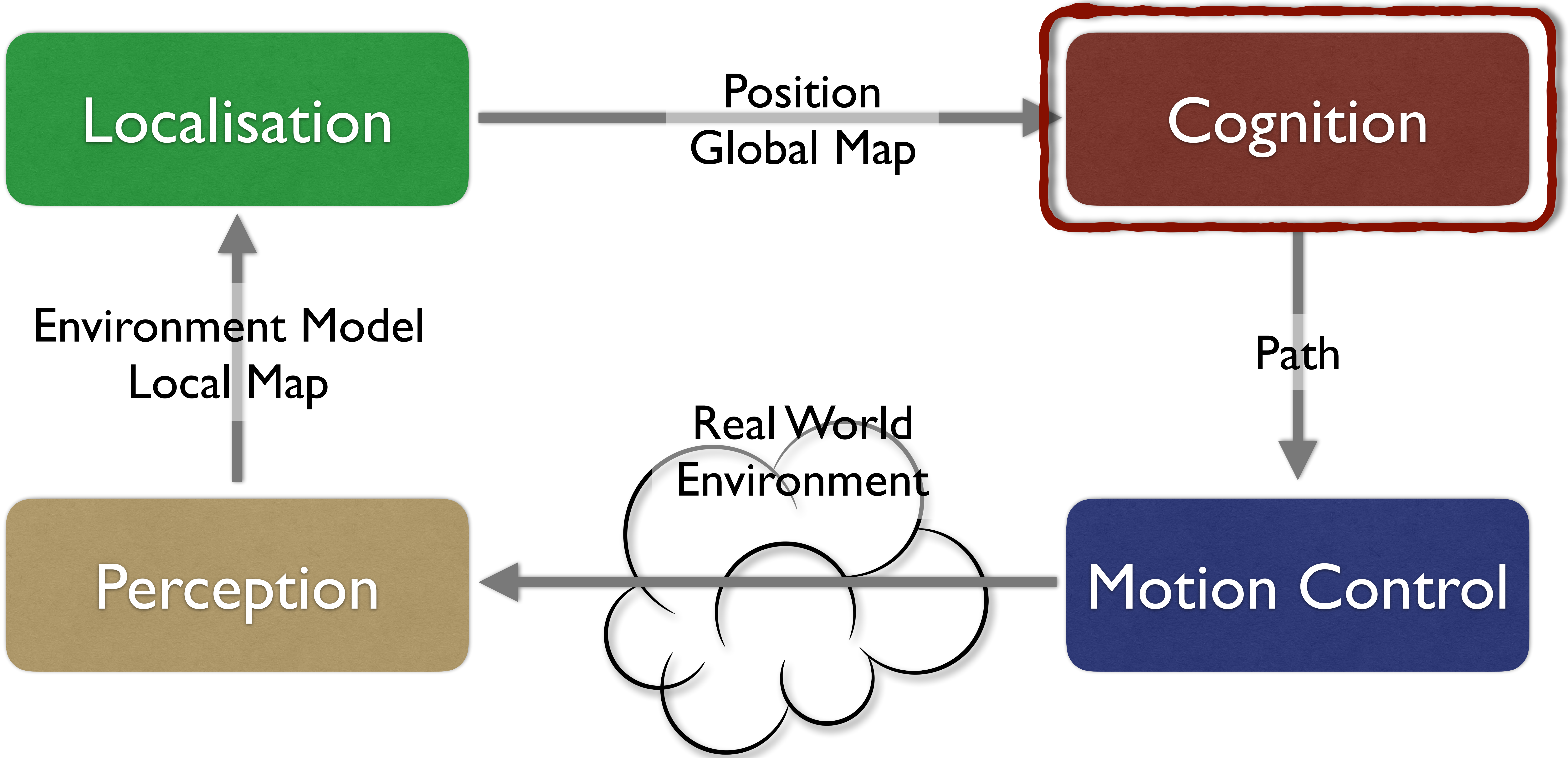
COMP329 Robotics and Autonomous Systems

Lecture 16: Beliefs, Desires and Intentions

Dr Terry R. Payne
Department of Computer Science
UNIVERSITY OF LIVERPOOL

General control architecture



Original Source: M. Wooldridge, S.Parsons, D.Grossi - updated by Terry Payne, Autumn 2016 & 2017

What is Practical Reasoning?

- Practical reasoning is reasoning directed towards actions — the process of figuring out what to do:

> *"... 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 and what the agent believes..." (Bratman)*

- Distinguish practical reasoning from ***theoretical reasoning***.

- Theoretical reasoning is directed towards beliefs.

Original Source: M. Wooldridge, S.Parsons, D.Grossi - updated by Terry Payne, Autumn 2016 & 2017

The components of Practical Reasoning

- Human practical reasoning consists of two activities:
  - ***deliberation***:
    - deciding ***what*** state of affairs we want to achieve
    - the outputs of deliberation are ***intentions***;
  - ***means-ends reasoning***:
    - deciding ***how*** to achieve these states of affairs
    - the outputs of means-ends reasoning are ***plans***.

Original Source: M. Wooldridge, S.Parsons, D.Grossi - updated by Terry Payne, Autumn 2016 & 2017

Intentions in Practical Reasoning

1. Intentions pose problems for agents, who need to determine ways of achieving them.

   If I have an intention to φ, you would expect me to devote resources to deciding how to bring about φ.

2. Intentions provide a "filter" for adopting other intentions, which must not conflict.

   If I have an intention to φ, you would not expect me to adopt an intention ψ that was incompatible with φ.

3. 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 φ fails, then all other things being equal, it will try an alternative plan to achieve φ.

Original Source: M. Wooldridge, S.Parsons, D.Grossi - updated by Terry Payne, Autumn 2016 & 2017

Intentions in Practical Reasoning

4. Agents believe their intentions are possible.

   That is, they believe there is at least some way that the intentions could be brought about.

5. Agents do not believe they will not bring about their intentions.

   It would not be rational of me to adopt an intention to φ if I believed I would fail with φ.

6. Under certain circumstances, agents believe they will bring about their intentions.

   If I intend φ, then I believe that under "normal circumstances" I will succeed with φ.

Original Source: M. Wooldridge, S.Parsons, D.Grossi - updated by Terry Payne, Autumn 2016 & 2017

Intentions in Practical Reasoning

7. Agents need not intend all the expected side effects of their intentions.

If I believe $\varphi \Rightarrow \psi$ and I intend that φ,

I do not necessarily intend ψ also.

- 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, and I may also intend to go to the dentist — but this does not imply that I intend to suffer pain!

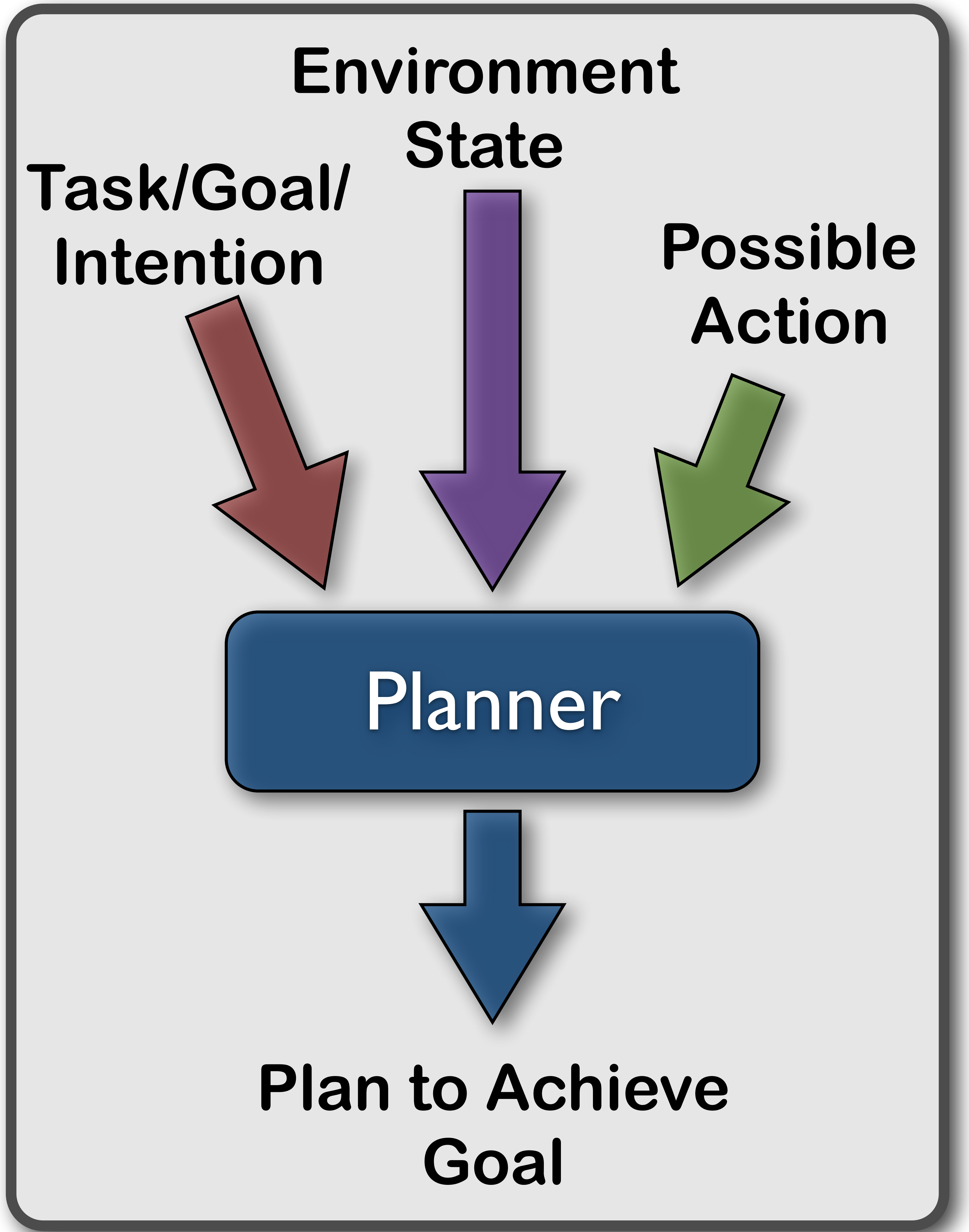
Original Source: M. Wooldridge, S.Parsons, D.Grossi - updated by Terry Payne, Autumn 2016 & 2017

Intentions are Stronger than Desire

> *"... 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..."*
>
> *Michael E. Bratman (1990)*

Means-ends Reasoning/Planning

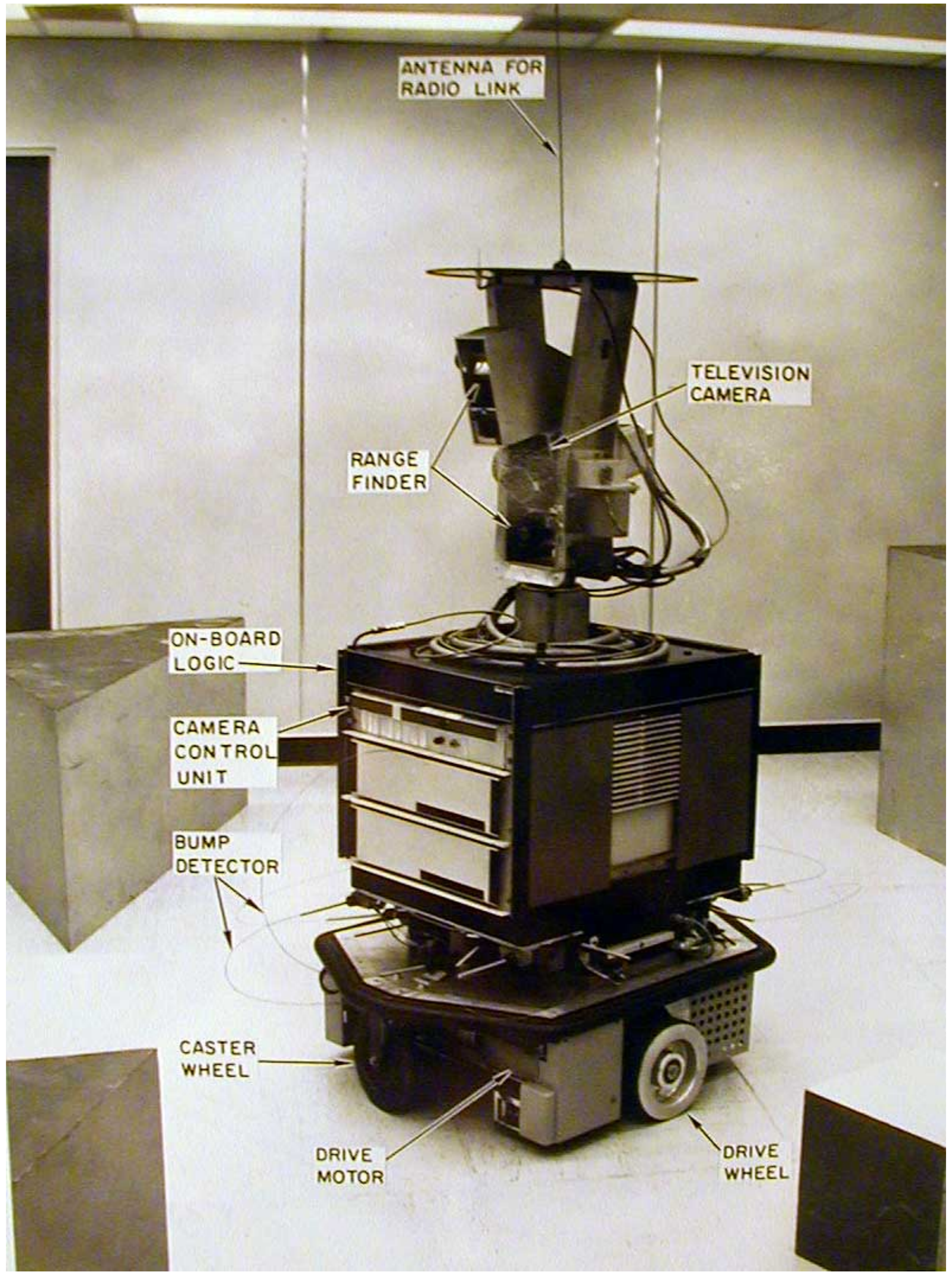
- Planning is the design of a course of action that will achieve some desired goal.
  - Basic idea is to give a planning system:
    - (representation of) goal/intention to achieve;
    - (representation of) actions it can perform;
    - (representation of) the environment;
  - and have it generate a ***plan*** to achieve the goal.
- This is ***automatic programming***.



Original Source: M. Wooldridge, S.Parsons, D.Grossi - updated by Terry Payne, Autumn 2016 & 2017

STRIPS Planner

- STRIPS
  - The **St**anford **R**esearch **I**nstitute **P**roblem **S**olver
  - Used by Shakey, the robot
    - Developed by Richard Fikes and Nils Nilsson in 1971 at SRI International



Original Source: M. Wooldridge, S.Parsons, D.Grossi - updated by Terry Payne, Autumn 2016 & 2017

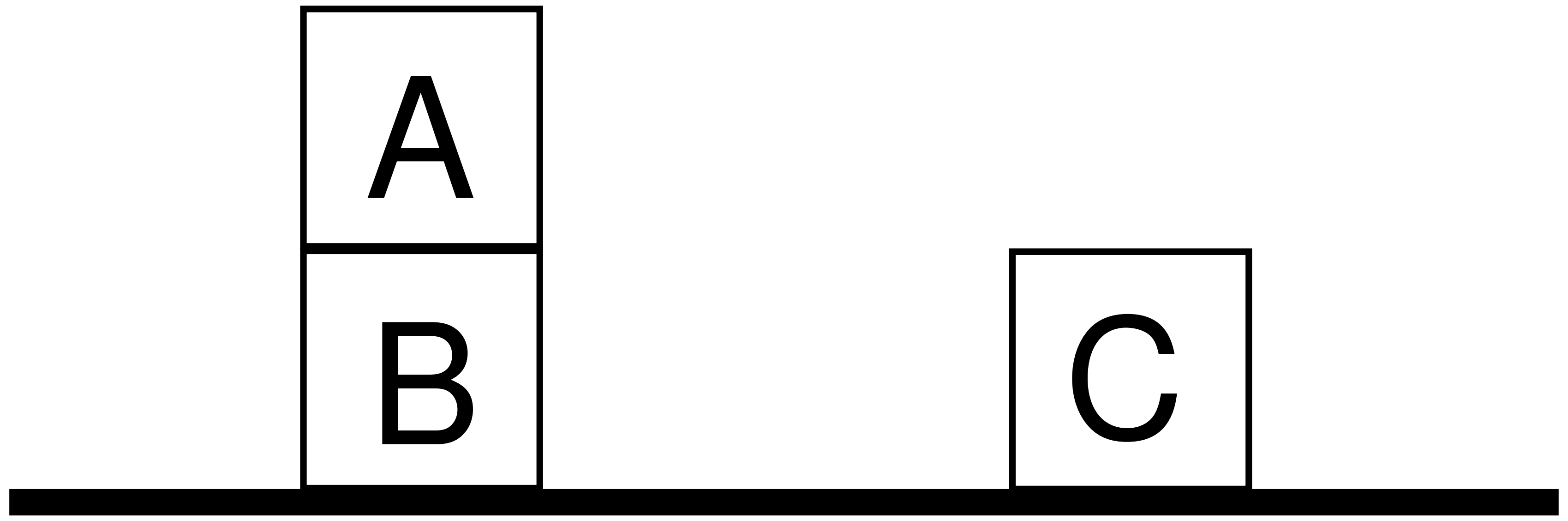
Representations

- ***Question***: How do we represent. . .
  - goal to be achieved;
  - state of environment;
  - actions available to agent;
  - plan itself.

- ***Answer***: We use ***logic***, or something that looks a lot like logic.

Blocksworld

- We'll illustrate the techniques with reference to the blocks world.
  - A simple (toy) world, in this case one where we consider toys



- The blocks world contains a robot arm, 3 blocks (A, B and C) of equal size, and a table-top.

- The aim is to generate a plan for the robot arm to build towers out of blocks.

Original Source: M. Wooldridge, S.Parsons, D.Grossi - updated by Terry Payne, Autumn 2016 & 2017

Blocksworld

- The environment is represented by an ontology.
- The closed world assumption is used
  - Anything not stated is assumed to be false.
- A goal is represented as a set of formulae.

Blocksworld Ontology

On(x,y)	object *x* on top of object *y*
OnTable(x)	object *x* is on the table
Clear(x)	nothing is on top of object *x*
Holding(x)	arm is holding *x*

Representation of the following blocks

Clear(A)
On(A, B)
OnTable(B)
Clear(C)
OnTable(C)
ArmEmpty

The goal:

{OnTable(A), OnTable(B), OnTable(C), ArmEmpty}

Original Source: M. Wooldridge, S.Parsons, D.Grossi - updated by Terry Payne, Autumn 2016 & 2017

Blocksworld Actions

- Each action has:
  - a ***name***: which may have arguments;
  - a ***pre-condition list***: list of facts which must be true for action to be executed;
  - a ***delete list***: list of facts that are no longer true after action is performed;
  - an ***add list***: list of facts made true by executing the action.
- Each of these may contain variables.
- What is a plan?
  - A sequence (list) of actions, with variables replaced by constants.

Original Source: M. Wooldridge, S.Parsons, D.Grossi - updated by Terry Payne, Autumn 2016 & 2017

Blocksworld Actions

$$Stack(x, y)$$

pre	$Clear(y) \wedge Holding(x)$
del	$Clear(y) \wedge Holding(x)$
add	$ArmEmpty \wedge On(x, y)$

*The **stack** action occurs when the robot arm places the object x it is holding is placed on top of object y.*

$$Pickup(x)$$

pre	$Clear(x) \wedge OnTable(x) \wedge ArmEmpty$
del	$OnTable(x) \wedge ArmEmpty$
add	$Holding(x)$

*The **pickup** action occurs when the arm picks up an object x from the table.*

$$UnStack(x, y)$$

pre	$On(x, y) \wedge Clear(x) \wedge ArmEmpty$
del	$On(x, y) \wedge ArmEmpty$
add	$Holding(x) \wedge Clear(y)$

*The **unstack** action occurs when the robot arm picks an object y up from on top of another object y.*

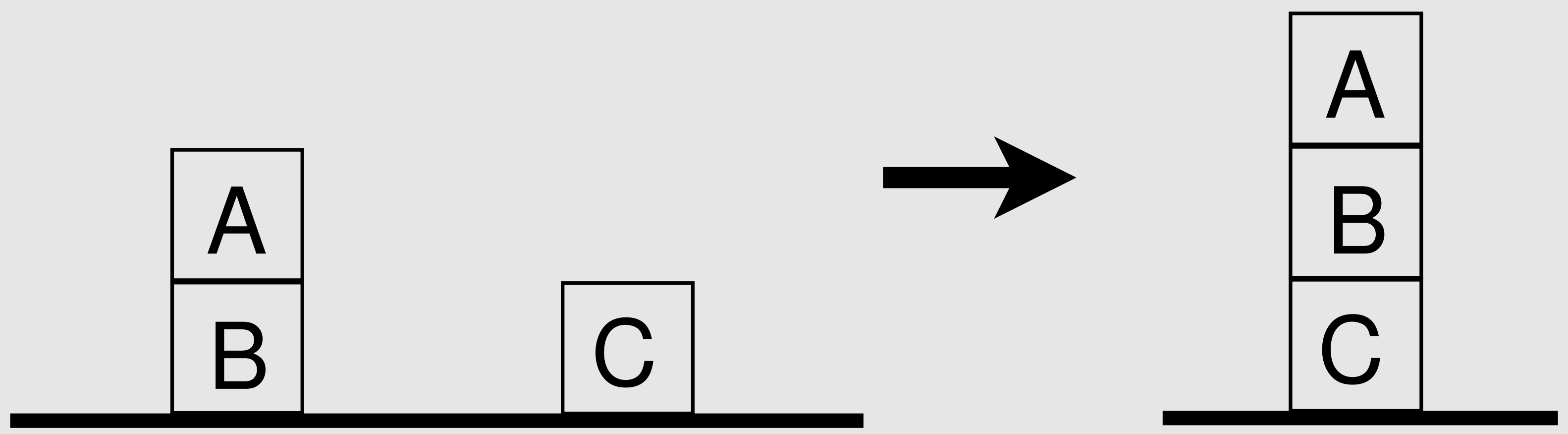
$$PutDown(x)$$

pre	$Holding(x)$
del	$Holding(x)$
add	$OnTable(x) \wedge ArmEmpty \wedge Clear(x)$

*The **putdown** action occurs when the arm places the object x onto the table.*

Using Plans

To get from here (left) to here (right)…



…we need this set of actions:

UnStack(A,B)

Putdown(A)

Pickup(B)

Stack(B, C)

Pickup(A)

Stack(A, B)

$$\begin{array}{ll} & Stack(x,y) \\ \text{pre} & Clear(y) \wedge Holding(x) \\ \text{del} & Clear(y) \wedge Holding(x) \\ \text{add} & ArmEmpty \wedge On(x,y) \end{array}$$

$$\begin{array}{ll} & UnStack(x,y) \\ \text{pre} & On(x,y) \wedge Clear(x) \wedge ArmEmpty \\ \text{del} & On(x,y) \wedge ArmEmpty \\ \text{add} & Holding(x) \wedge Clear(y) \end{array}$$

$$\begin{array}{ll} & Pickup(x) \\ \text{pre} & Clear(x) \wedge OnTable(x) \wedge ArmEmpty \\ \text{del} & OnTable(x) \wedge ArmEmpty \\ \text{add} & Holding(x) \end{array}$$

$$\begin{array}{ll} & PutDown(x) \\ \text{pre} & Holding(x) \\ \text{del} & Holding(x) \\ \text{add} & OnTable(x) \wedge ArmEmpty \wedge Clear(x) \end{array}$$

Formal Representation

- Let's relate the STRIPS model back to the formal description of an agent we talked about before.
  - This will help us to see how it fits into the overall picture.

- As before we assume that the agent has a set of actions Ac, and we will write individual actions as α_1, α_2 and so on.

- Now the actions have some structure, each one has preconditions $P_{\alpha i}$, add list $A_{\alpha i}$, and delete list $D_{\alpha i}$, for each $\alpha_i \in Ac$:

$$\alpha_i = \langle P_{\alpha_i}, D_{\alpha_i}, A_{\alpha_i} \rangle$$

- A plan is just a sequence of actions, where each action is one of the actions from Ac:

$$\pi = (\alpha_1, \ldots, \alpha_n)$$

Formal Representation

- A planning problem is therefore: $\langle B_0, Ac, I\rangle$
  - B_0 is the set of beliefs the agent has about the world.
  - Ac is the set of actions, and
  - I is a goal (or intention)

- Since actions change the world, any rational agent will change its beliefs about the world as a result of carrying out actions.
  - Thus, a plan π for a given planning problem will be associated with a sequence of sets of beliefs:

$$B_0 \stackrel{\alpha_1}{\to} B_1 \stackrel{\alpha_2}{\to} \cdots \stackrel{\alpha_n}{\to} B_n$$

  - In other words at each step of the plan the beliefs are updated by removing the items in the delete list of the relevant action and adding the items in the add list.

Original Source: M. Wooldridge, S.Parsons, D.Grossi - updated by Terry Payne, Autumn 2016 & 2017

Formal Representation

- A plan π is said to be acceptable with respect to the problem $\langle B_0, Ac, I \rangle$ if and only if, for all $1 \leq j \leq n$, $B_{j-1} \models P_{\alpha j}$
  - In other words, the pre-requisites for each action have to be true right before the action is carried out.
    - We say this because the pre-conditions don't have to be in B_{j-1}, we just have to be able to prove the pre-conditions from B_{j-1}.

- A plan π is correct if it is acceptable, and: $B_n \models i$
  - In other words, it is correct if it is acceptable and the final state makes the goal true.

Original Source: M. Wooldridge, S.Parsons, D.Grossi - updated by Terry Payne, Autumn 2016 & 2017

Implementing Practical Reasoning Agents

- A first pass at an implementation of a practical reasoning agent:
- For now we will not be concerned with stages 2 or 3.
  - These are related to the functions *see* and *next* from Lectures 2-3 on Agents.

```
Agent Control Loop Version 1
1.  while true
2.      observe the world;
3.      update internal world model;
4.      deliberate about what intention
          to achieve next;
5.      use means-ends reasoning to get
          a plan for the intention;
6.      execute the plan
7.  end while
```

Implementing Practical Reasoning Agents

- see is as before:
  - `see`: E → Percept

- Instead of the function `next`...
  - which took a percept and used it to update the internal state of an agent

- …we have a belief revision function:
  - `brf`: $\mathscr{P}$\{Bel\} x Percept → $\mathscr{P}$\{Bel\}
    - $\mathscr{P}$\{Bel\} is the power set of beliefs
    - Bel is the set of all possible beliefs that an agent might have.

```
Agent Control Loop Version 1
1.  while true
2.      observe the world;
3.      update internal world model;
4.      deliberate about what intention
          to achieve next;
5.      use means-ends reasoning to get
          a plan for the intention;
6.      execute the plan
7.  end while
```

Implementing Practical Reasoning Agents

- Problem: deliberation and means-ends reasoning processes are not instantaneous.
  - They have a ***time cost***.
- Suppose that deliberation is ***optimal*** in that if it selects some intention to achieve, then this is the best thing for the agent.
  - i.e. it maximises expected utility.
- So the agent selects an intention to achieve that would have been optimal ***at the time it observed the world***.
  - This is calculative rationality.
- The ***world may change*** in the meantime.
  - Even if the agent can compute the right thing to do, it may not do the right thing.
  - Optimality is hard.

Original Source: M. Wooldridge, S.Parsons, D.Grossi - updated by Terry Payne, Autumn 2016 & 2017

Implementing Practical Reasoning Agents

- Let's make the algorithm more formal with the algorithm opposite
  - where $I \subseteq Int$, i.e the set of intentions,
  - `plan()` is exactly what we discussed above,
  - `brf()` is the belief revision function,
  - and `execute()` is a function that executes each action in a plan.

```
Agent Control Loop Version 2
1.  B := B_0; /* initial beliefs */
2.  while true do
3.      get next percept ρ;
4.      B := brf(B, ρ);
5.      I := deliberate(B);
6.      π := plan(B, I);
7.      execute(π)
8.  end while
```

- How might we implement these functions?

Original Source: M. Wooldridge, S.Parsons, D.Grossi - updated by Terry Payne, Autumn 2016 & 2017

Deliberation

- How does an agent ***deliberate***?
  - begin by trying to understand what the ***options*** available to you are;
  - ***choose*** between them, and ***commit*** to some.

- Chosen options are then ***intentions***.

- The deliberate function can be decomposed into two distinct functional components:
  - ***option generation***; and
  - ***filtering***.

Original Source: M. Wooldridge, S.Parsons, D.Grossi - updated by Terry Payne, Autumn 2016 & 2017

Option Generation and Filtering

- Option Generation
  - In which the agent generates a set of possible alternatives
  - 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
    - ***desires***
- 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.
    - ***intentions***

Implementing Practical Reasoning Agents

Agent Control Loop Version 3

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

Original Source: M. Wooldridge, S.Parsons, D.Grossi - updated by Terry Payne, Autumn 2016 & 2017

Under Commitment

"... Some time in the not-so-distant future, you are having trouble with your new household robot. You say "Willie, bring me a beer." The robot replies "OK boss." Twenty minutes later, you screech "Willie, why didn't you bring me that beer?" It answers "Well, I intended to get you the beer, but I decided to do something else." Miffed, you send the wise guy back to the manufacturer, complaining about a lack of commitment..."

P. R. Cohen and H. J. Levesque (1990). Intention is choice with commitment. Artificial intelligence, 42(2), 213-261.

Over Commitment

"...After retrofitting, Willie is returned, marked "Model C: The Committed Assistant." Again, you ask Willie to bring you a beer. Again, it accedes, replying "Sure thing." Then you ask: "What kind of beer did you buy?" It answers: "Genessee." You say "Never mind." One minute later, Willie trundles over with a Genessee in its gripper..."

P. R. Cohen and H. J. Levesque (1990). Intention is choice with commitment. Artificial intelligence, 42(2), 213-261.

Wise Guy ???

"…After still more tinkering, the manufacturer sends Willie back, promising no more problems with its commitments. So, being a somewhat trusting customer, you accept the rascal back into your household, but as a test, you ask it to bring you your last beer. [. . .]

The robot gets the beer and starts towards you. As it approaches, it lifts its arm, wheels around, deliberately smashes the bottle, and trundles off. Back at the plant, when interrogated by customer service as to why it had abandoned its commitments, the robot replies that according to its specifications, it kept its commitments as long as required — commitments must be dropped when fulfilled or impossible to achieve. By smashing the bottle, the commitment became unachievable…"

P. R. Cohen and H. J. Levesque (1990). Intention is choice with commitment. Artificial intelligence, 42(2), 213-261.

Degrees of Commitment

- 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 as long as it is still believed possible.

Original Source: M. Wooldridge, S.Parsons, D.Grossi - updated by Terry Payne, Autumn 2016 & 2017

Degrees of Commitment

- An agent has commitment both to:
  - ***ends*** (i.e., the state of affairs it wishes to bring about), and
  - ***means*** (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.
  - However, to write the algorithm down we need to refine our notion of plan execution.

Original Source: M. Wooldridge, S.Parsons, D.Grossi - updated by Terry Payne, Autumn 2016 & 2017

Degrees of Commitment

- If π is a plan, then:
  - *empty(π)* is true if there are no more actions in the plan.
  - *hd(π)* returns the first action in the plan.
  - *tail(π)* returns the plan minus the head of the plan.
  - *sound(π, I, B)* means that π is a correct plan for I given B.

Agent Control Loop Version 4

```
B := B_0;
I := I_0;
while true do
    get next percept ρ;
    B := brf(B, ρ);
    D := options(B, I);
    I := filter(B, D, I);
    π := plan(B, I);
    while not empty(π) do
        α := hd(π);
        execute(α);
        π := tail(π);
        get next percept ρ;
        B := brf(B, ρ);
        if not sound(π, I, B) then
            π := plan(B, I)
        end-if
    end-while
end-while
```

Single Minded Commitment

- 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.***

```
Agent Control Loop Version 5

1.  B := B_0;
2.  I := I_0;
3.  while true do
4.      get next percept ρ;
5.      B := brf(B, ρ);
6.      D := options(B, I);
7.      I := filter(B, D, I);
8.      π := plan(B, I);
9.      while not empty(π)
                or succeeded(I, B)
                or impossible(I, B)) do
10.         α := hd(π);
11.         execute(α);
12.         π := tail(π);
13.         get next percept ρ;
14.         B := brf(B, ρ);
15.         if not sound(π, I, B) then
16.             π := plan(B, I)
17.         end-if
18.     end-while
19. end-while
```


Original Source: M. Wooldridge, S.Parsons, D.Grossi - updated by Terry Payne, Autumn 2016 & 2017

Intention 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; or
  - it believes it has achieved its current intentions; or
  - 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.

```
Agent Control Loop Version 6

1.  B := B_0;
2.  I := I_0;
3.  while true do
4.      get next percept ρ;
5.      B := brf(B, ρ);
6.      D := options(B, I);
7.      I := filter(B, D, I);
8.      π := plan(B, I);
9.      while not empty(π)
                or succeeded(I, B)
                or impossible(I, B)) do
10.         α := hd(π);
11.         execute(α);
12.         π := tail(π);
13.         get next percept ρ;
14.         B := brf(B, ρ);
15.         D := options(B, I);
16.         I := filter(B, D, I);
17.         if not sound(π, I, B) then
18.             π := plan(B, I)
19.         end-if
20.     end-while
21. end-while
```

Intention Reconsideration

- 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 attentions 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.

```
Agent Control Loop Version 7

1.  B := B_0;
2.  I := I_0;
3.  while true do
4.      get next percept \rho;
5.      B := brf(B, \rho);
6.      D := options(B, I);
7.      I := filter(B, D, I);
8.      \pi := plan(B, I);
9.      while not empty(\pi)
                or succeeded(I, B)
                or impossible(I, B)) do
10.         \alpha := hd(\pi);
11.         execute(\alpha);
12.         \pi := tail(\pi);
13.         get next percept \rho;
14.         B := brf(B, \rho);
15.         if reconsider(I, B) then
16.             D := options(B, I);
17.             I := filter(B, D, I);
18.         end-if
19.         if not sound(\pi, I, B) then
12.             \pi := plan(B, I)
21.         end-if
22.     end-while
23. end-while
```

Intention Reconsideration

- The possible interactions between meta-level control and deliberation are:

Situation number	Chose to deliberate?	Changed intentions?	Would have changed intentions?	$reconsider(\ldots)$ optimal?
1	No	—	No	Yes
2	No	—	Yes	No
3	Yes	No	—	No
4	Yes	Yes	—	Yes

- An important assumption: cost of **reconsider**(. . .) is ***much*** less than the cost of the deliberation process itself.

Intention Reconsideration

Situation number	Chose to deliberate?	Changed intentions?	Would have changed intentions?	$reconsider(\ldots)$ optimal?
1	No	—	No	Yes
2	No	—	Yes	No
3	Yes	No	—	No
4	Yes	Yes	—	Yes

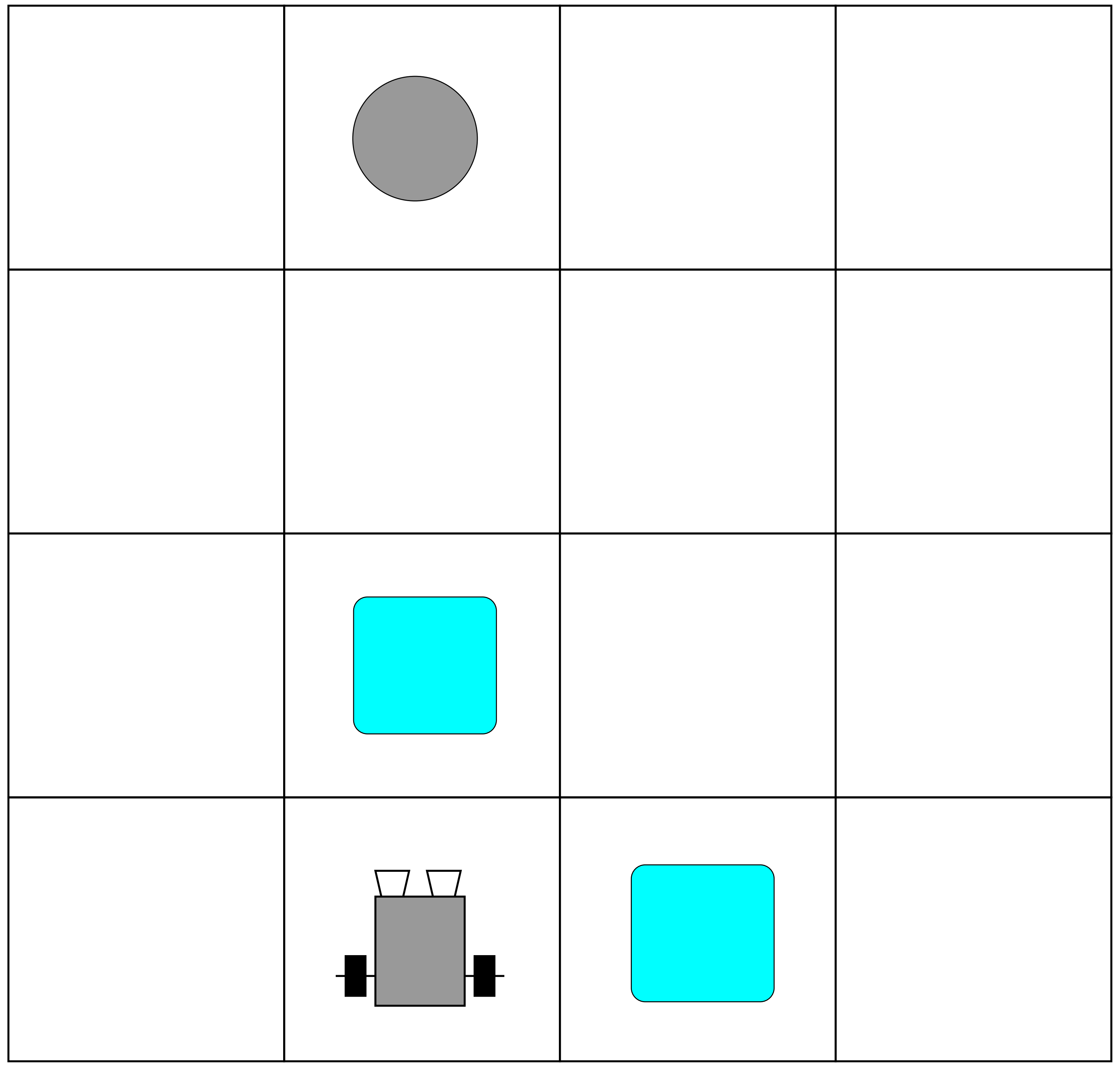
```
if reconsider(I, B) then
    D := options(B, I);
    I := filter(B, D, I);
end-if
```

- In situation (1), the agent did not choose to deliberate, and as a consequence, did not choose to change intentions. Moreover, if it ***had*** chosen to deliberate, it would not have changed intentions. In this situation, the `reconsider(...)` function is behaving optimally.
- In situation (2), the agent did not choose to deliberate, but if it had done so, it ***would*** have changed intentions. In this situation, the `reconsider(...)` function is ***not*** behaving optimally.
- In situation (3), the agent chose to deliberate, but did not change intentions. In this situation, the `reconsider(...)` function is ***not*** behaving optimally.
- In situation (4), the agent chose to deliberate, and did change intentions. In this situation, the `reconsider(...)` function is behaving optimally.

Original Source: M. Wooldridge, S.Parsons, D.Grossi - updated by Terry Payne, Autumn 2016 & 2017

Optimal Intention 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, and
    - ***cautious*** agents: stop to reconsider after every action.
- ***Dynamism*** in the environment is represented by the rate of world change, γ.
  - Experiments were carried out using Tileword.

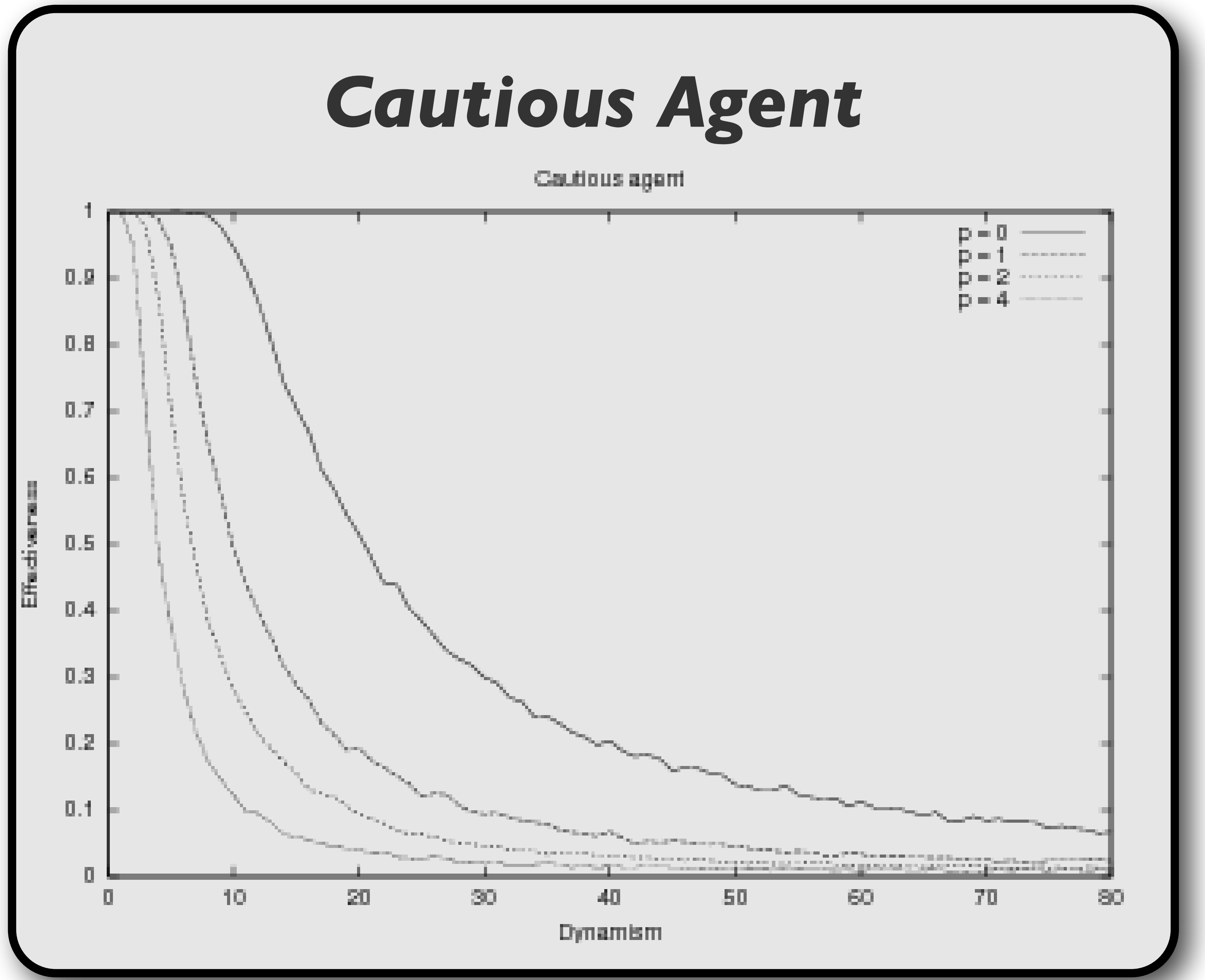


Original Source: M. Wooldridge, S.Parsons, D.Grossi - updated by Terry Payne, Autumn 2016 & 2017

Optimal Intention Reconsideration

- If γ is ***low*** (i.e., the environment does not change quickly), then bold agents do well compared to cautious ones.
  - This is because cautious ones waste time reconsidering their commitments while bold agents are busy working towards — and achieving — their intentions.
- If γ is ***high*** (i.e., the environment changes frequently), then cautious agents can outperform bold agents.
  - This is because they are able to recognise when intentions are doomed, and also to take advantage of serendipitous situations and new opportunities when they arise.
- When planning costs are high, this advantage can be eroded.

low ⟵ γ ⟶ high



Original Source: M. Wooldridge, S.Parsons, D.Grossi - updated by Terry Payne, Autumn 2016 & 2017

Summary

- This lecture has covered a lot of ground on practical reasoning.
  - We started by discussing what practical reasoning was, and how
    it relates to intentions.
  - We then looked at planning (how an agent achieves its desires) and how deliberation and means-ends reasoning fit into the basic agent control loop.
  - We then refined the agent control loop, considering commitment and intention reconsideration.
  - Finally, we looked at an implemented system (the textbook discusses a couple of others).