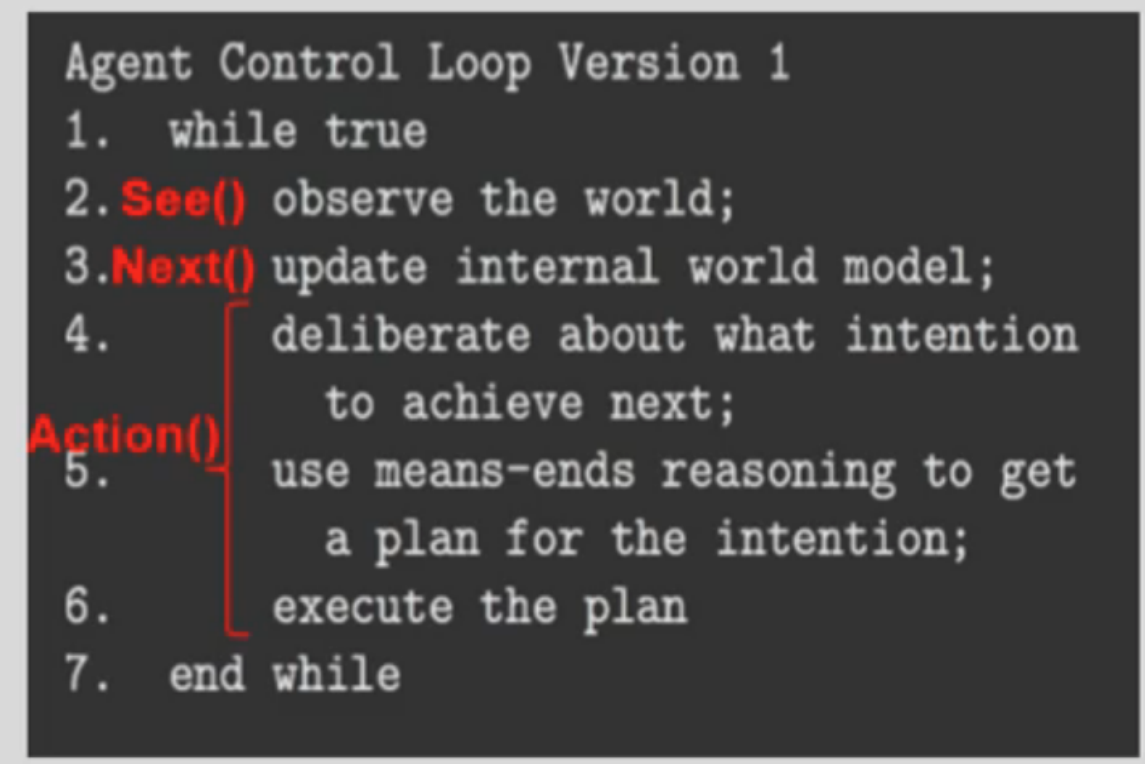
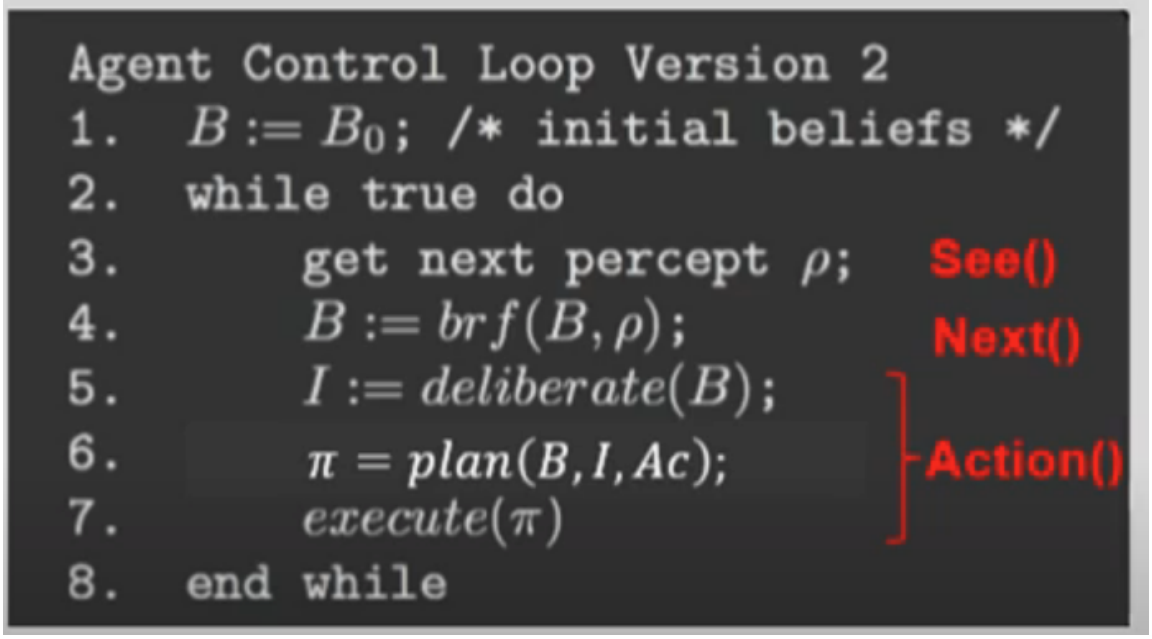
Implementation of Agents

▼ Implementation of Agent Control Loop

This is the pseudo code for the ACL



We can make it more formal



We add the beliefs of the plan (line 1), the percept rho (line 3), and we use the belief revision function in line 4 to update our belief of the world. We deliberate what our next actions are going to be using those beliefs in line 5 and generate a plan on line 6, and in line 7, we execute the plan.

The deliberate function is part of the action function in the abstract ACL. It receives our beliefs of the world and generates an intention. It has two steps:

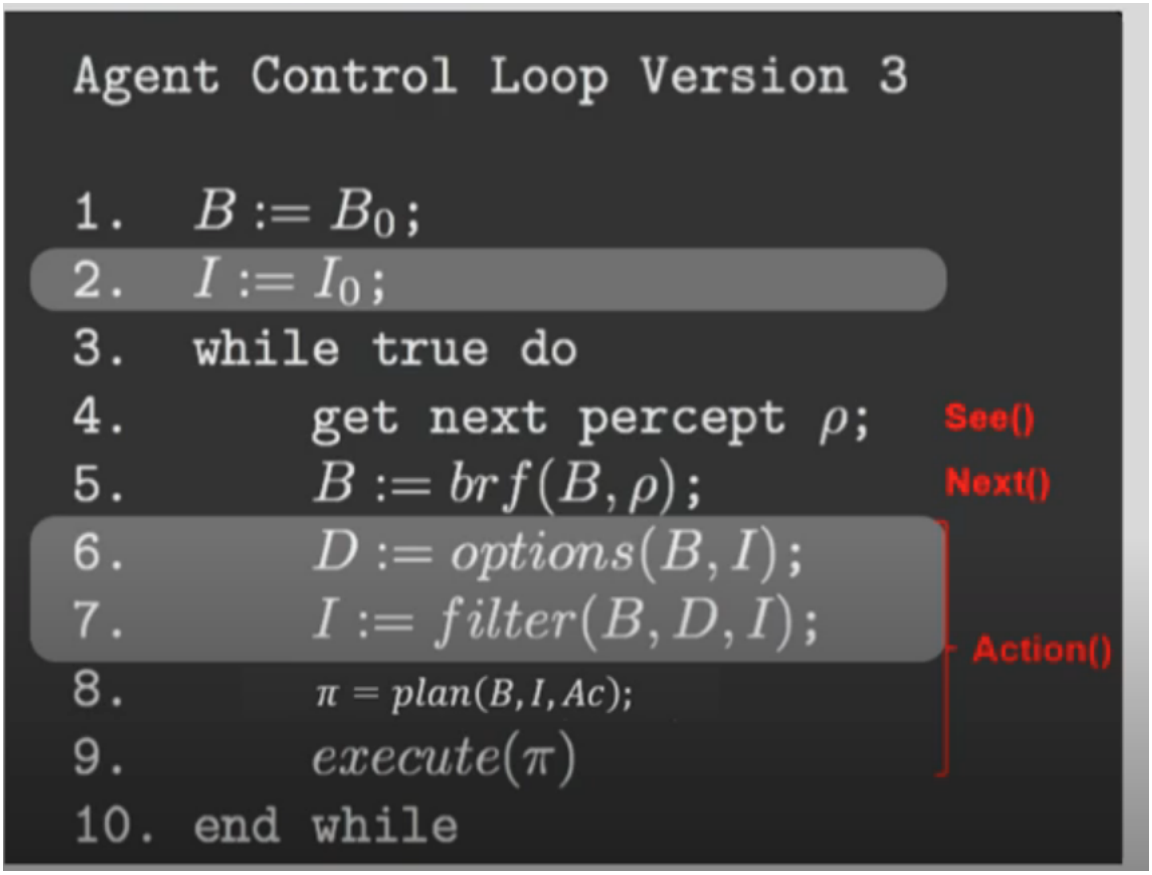
1. Given our beliefs of the world, what possible intentions could we have?
2. Choose between them and commit to it.

This relates to the coalitions and bidding strategies we previously studied.

The first part is the Auction Generation: generating the auctions that are available to the agent (what states can I reach or plan to get to).

The second part is the Filtering. We filter our desires with our beliefs and current intentional state to generate a new intentional state and update it if needed.

We keep modifying the loop by adding the following



Now in line 2 we have an initial intention, and in line 6 we take in our beliefs and current intention and generate the options that are available to us and our desires. In line 7 we filter the beliefs, desires and intentions to update our intention if necessary. these two functions together are known as deliberation.

▼ Commitment

Under commitment or over commitment can be an issue. If you ask something to an under committed agent, it might drop it's intention of carrying out that order and do something else instead. On the other hand, if you ask something to an over committed agent, you cannot withdraw that order, which is not ideal either.

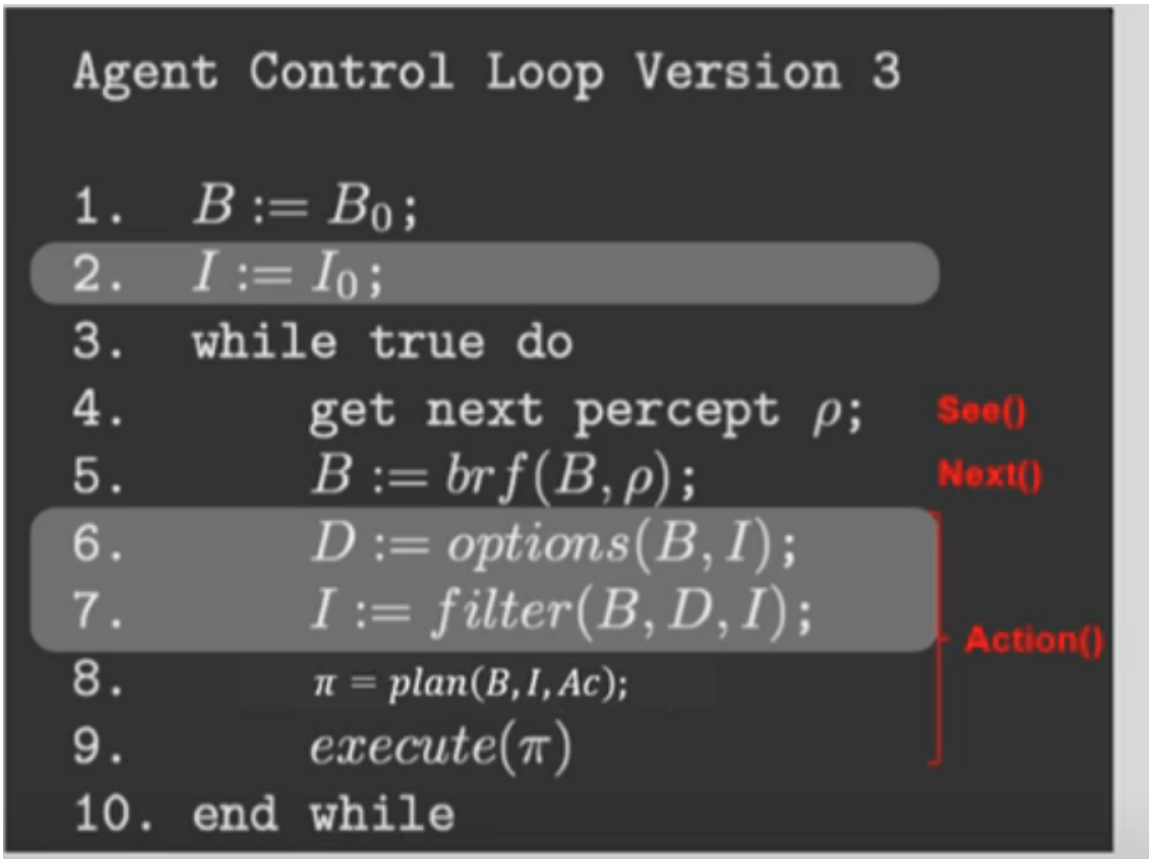
We want a balance point between the two, in which an agent could drop it's intention in case it was't possible (like asking them for a beer and it being smashed).

We define multiple types of commitment:

- Blind Commitment: an agent maintains an intention until it believes it's been achieved.
- Single Minded Commitment: an agent maintains an intention either until it's achieved or it doesn't think it's possible
- Open Minded Commitment: an agent maintains an intention while it believes it's possible but it's easy for it to go to other intentions as well

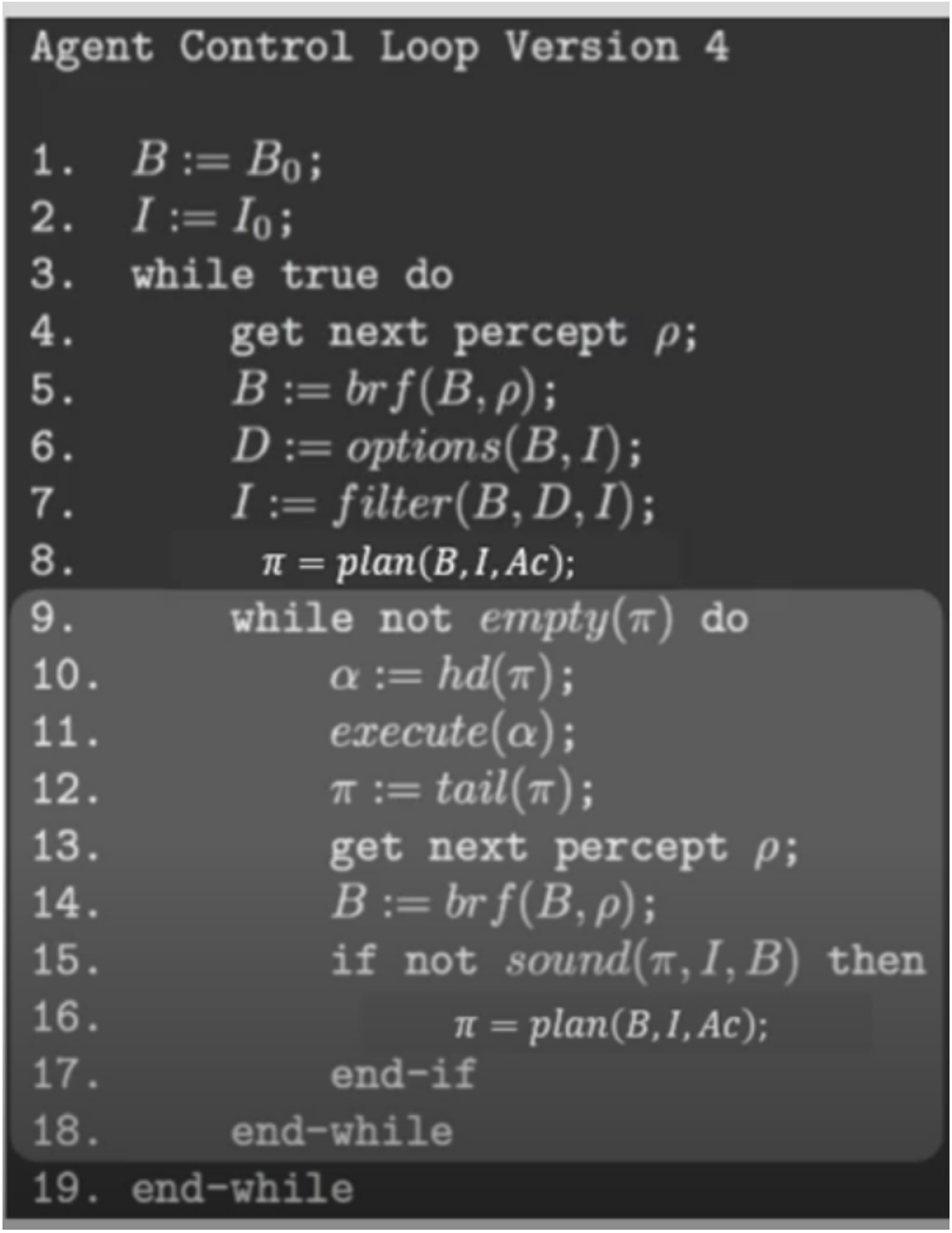
An agent has to be committed to it's **means** (plan) and to it's **ends** (intention).

If we look again at out current ACL



We update our initial intention at line 7 if we need to, then generate and execute a plan without any chance to revise it. The plan only gets looked at once. This is blind commitment.

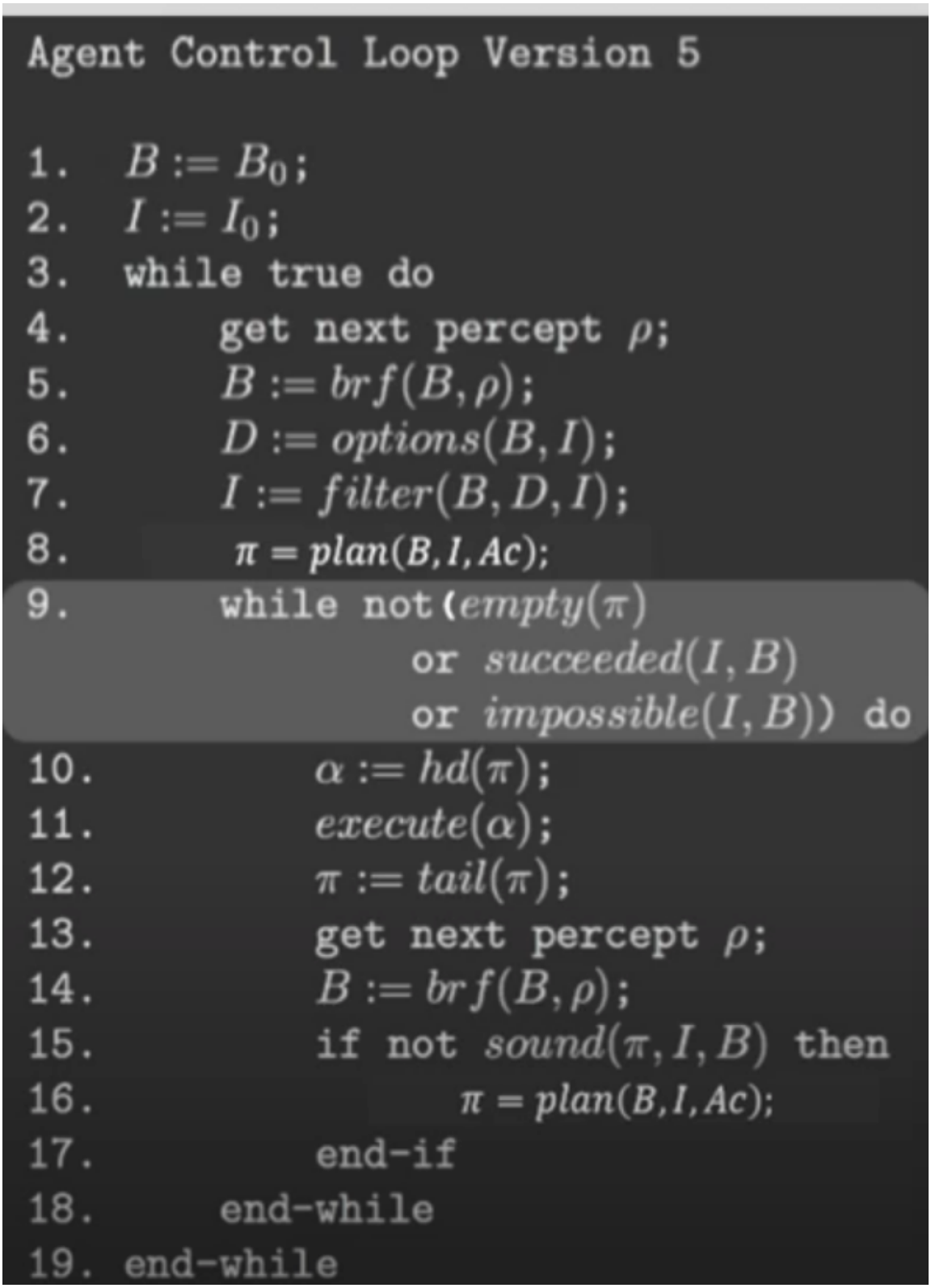
We can fix this by adding another nested while loop



This makes the ACL much more reactive, checking the soundness of the plan after every single action. The head function gets the first action and executes it, and the tail function removes that first item from the plan, giving us the remainder of it. After we do every action, we check for the soundness of the plan and update that plan if it's not longer sound or possible.

This means the agent is not committed to the plan anymore. However, we are still committed to the intention we had. Because of this, the agent has **single minded commitment**.

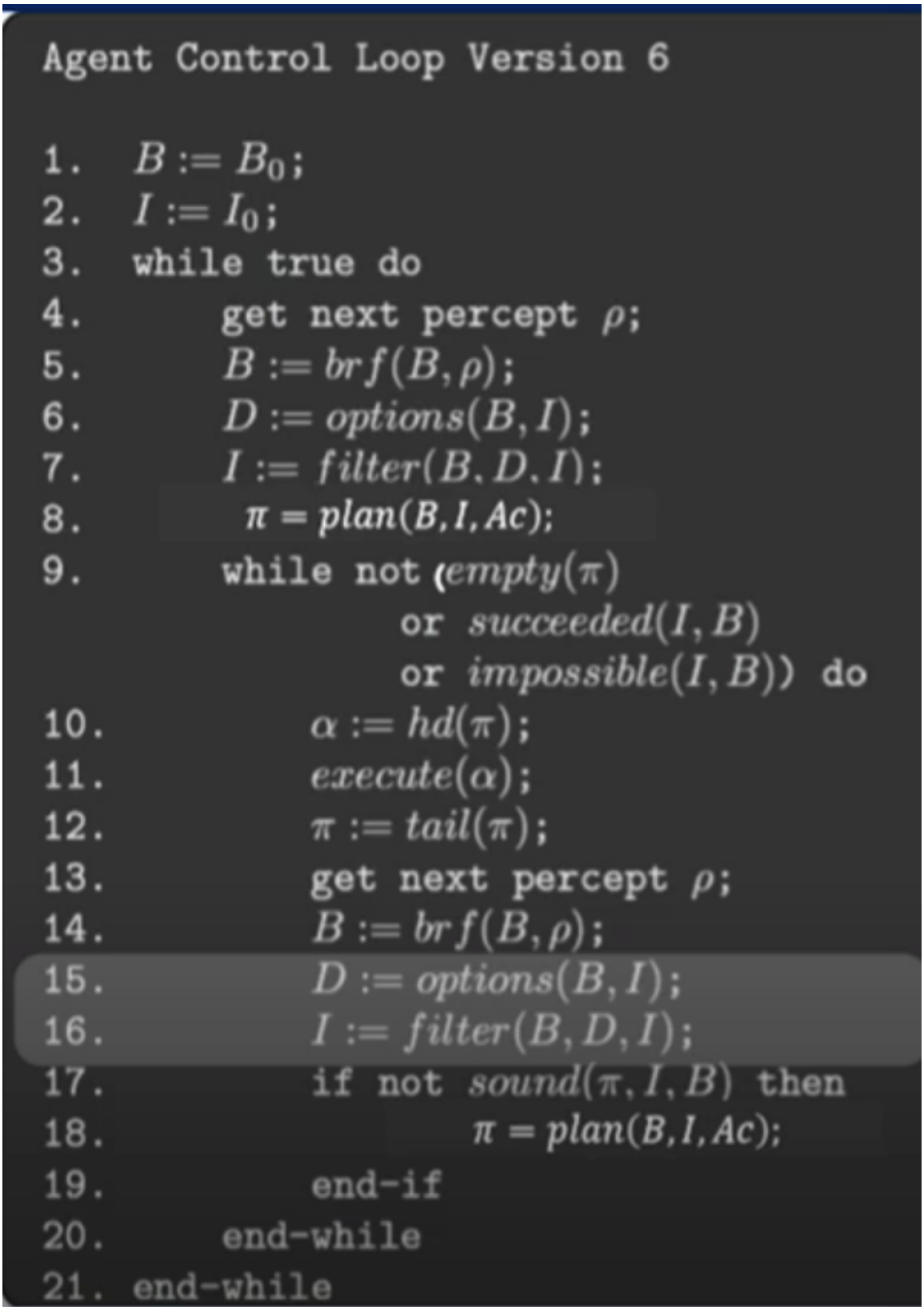
We can make the agent reconsider it's intentions by adding some extra conditions in line 9



Now we check the intention and reconsiders it if needed, because as soon as we exit that while loop, we will reconsider the plan and the intention. We make the agent reconsider it's intention either after executing the plan, after deciding the plan is impossible or after deciding the intention has

been achieved. This means the agent now has an **open minded commitment**.

We can add even more reconsideration. By adding the following

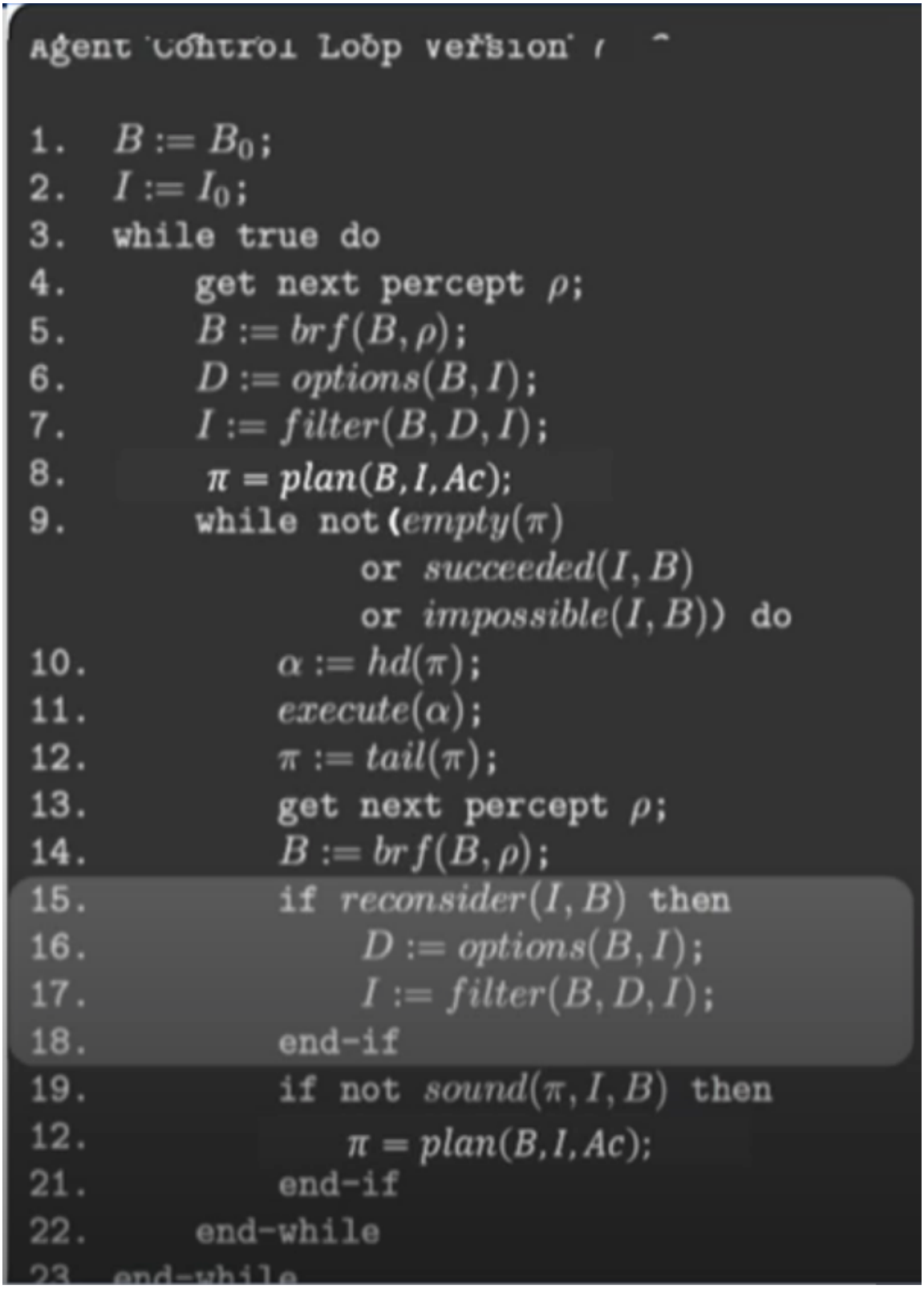


```
Agent Control Loop Version 6

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, Ac);
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, Ac);
19.         end-if
20.     end-while
21. end-while
```

we make the agent reconsider it's intention after every action. This reconsideration, however, can be very costly.

We can fix that by adding a meta-level construct that decides wether or not the agent should reconsider



We assume reconsider() executes in less time than options() and filter(), saving time in the long run.

Now we are not committed to our plan or intention, checking them after every action executed. This makes it a very reactive agent. A truly goal-driven agent.