Active Perception Within BDI Agents Reasoning Cycle

Gustavo R. Silva
Department of Automation and Systems
Universidade Federal de Santa Catarina
gustavorezendesilva@hotmail.com

Jomi F. Hübner
Department of Automation and Systems
Universidade Federal de Santa Catarina
jomi.hubner@ufsc.br

Leandro B. Becker
Department of Automation and Systems
Universidade Federal de Santa Catarina
leandro.becker@ufsc.br

ABSTRACT

In multi-agent systems the main process responsible for obtaining information about the environment is perception, generally this process is performed passively regardless the agent's intentional state. However, especially when inserted in the real world, a frequent problem is that agents have partial perception of the environment, failing to perceive some relevant information. To circumvent this problem, a solution is to actively take actions to perceive what is of interest to the agent, for example, in a computer vision system, the camera can be repositioned to have a better view of an object. This work aims to develop an active perception model integrated with the reasoning cycle of BDI agents. Experiments are performed using BDI agents with ROS to command unmanned aerial vehicles to analyze the benefits and impacts of using cognitive agents with active perception to program robot intelligence.

KEYWORDS

Active Perception; BDI agents; UAVs

ACM Reference Format:
Gustavo R. Silva, Jomi F. Hübner, and Leandro B. Becker. 2021. Active Perception Within BDI Agents Reasoning Cycle. In *Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), Online, May 3–7, 2021*, IFAAMAS, 8 pages.

1 INTRODUCTION

One of the mechanisms that agents have to gather information about the state of the environment is the perception, which is responsible for sensing the world and translating these information into high-level abstractions suitable for agent deliberation. As proposed in [9], perception can be classified into two different types, the bottom-up (passive) or top-down (active) approach. *Passive perception* is described as a process that does not require the agent to deliberate about its sensing needs, it perceives the environment in the same way independently of its own internal state. *Active perception* was characterized by being goal-driven, which means that the goals of the agent determine what needs to be sensed and then a proper action is taken in order to perceive what is needed.

In the usual BDI agent model passive perception is taken into account [7, 12]. The regular BDI architecture assumes that the agent knowledge is updated. However, it is possible that agents have partial or outdated information about the environment, specially when they are inserted in the real world. Therefore, it would be advantageous for the agents to update or acquire knowledge about the world before deciding their actions.

This work proposes a model for active perception integrated with the BDI architecture. Since the advantages of active perception are highlighted in complex real world scenarios, the proposed model is evaluated in real applications. Therefore, the active perception mechanism developed is evaluated using simulated unmanned aerial vehicles (UAVs) as testbed.

The reminder parts of this paper are organized as follows. Section 2 provides the literature review for what is discussed in this work; Section 3 describes the proposed active perception model; Section 4 details some design aspects; Section 5 describes a possible implementation for the active perception mechanism; Section 6 reports the result of our experiments; and Section 7 outlines the conclusions and future works.

2 RELATED WORK

In this work, active perception is reviewed in the context of intelligent agents. Firstly, the review will address works that have a more theoretical point of view, such as [9, 11], then it will concentrate on practical works, like [1, 2, 5, 6, 10].

In [11] the authors point out the lack of theories and general models for perception in MAS, despite its importance, highlighting that most MAS adopt a simplistic model for perception or *ad hoc* solutions. Therefore, they proposed a generic model for active perception in situated MAS, which is composed of three functional modules: sensing, interpreting, and filtering.

Sensing is the process of mapping the state of the environment to a representation, which depends on two factors: *foci*, and *perceptual laws*. The latter is the set of environmental constraints of the representation, e.g. something behind an obstacle can not be perceived, in the physical world the *perceptual laws* are intrinsic to the environment. The former, *foci*, is the direction of perception, which allows the agent to choose what type of information it wishes to perceive, e.g. an agent can select to smell or see. Following, *interpreting* is the translation of a representation into a perception, which is an expression that is understandable by the machinery of the agent. Lastly, *filtering* is the process of selecting only the perceptions that match a specific criteria. The authors make the following comparison to biological systems: "Focus selection can be viewed as choosing a particular sense to observe the environment, while filter selection is comparable to the direction of attention, both driven by the current interests".

This model of active perception is interesting since it allows the agent to direct its focus to relevant aspects of the environment. However, it considers that what the agents want to perceive is

Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), U. Endriss, A. Nowé, F. Dignum, A. Lomuscio (eds.), May 3–7, 2021, Online.

directly available, which in physical systems may not be true, e.g. using a visual focus the object of interest may be out of range. One solution for this problem would be to include a mechanism in the *foci* step to ensure that the agent is able to perceive what it wishes, e.g. the visual system could be repositioned in order to be in range of the object of interest.

In [9], the authors emphasize the importance of the perception process for intelligent agents, since it is the main mechanism used by agents to gather knowledge about the environment, and most of the decisions taken by the agents are based on what it knows about the environment. Then, it is pointed out that the definition of an agent's perception (sensing) behaviour usually consists on defining the strategies for two factors: *dynamism*, frequency of sensing, and *selectivity*, choosing what to sense. The authors argue that the answer for the sensing behaviour lies in active (goal-driven) perception, and that situational awareness (SA) is an appropriate approach for solving active perception. Therefore, they proposed a SA mechanism that enables the agent to switch between goal-driven and data-driven behaviour, where the top-down goal-driven process works by projecting what is known about the environment into the near future revealing what must be sensed, in the case of BDI which beliefs must be updated. One missing point of this work is that it does not address the problem of integrating it within any agent's architecture.

In [10], an active perception framework was developed in order to enable an autonomous car to redirect its sensors to focus on relevant surrounding area, in urban traffic scenarios. To accomplish this, three main criteria are taken into account, the importance of other vehicles, the available information about different vehicles, and the sensor coverage of the vehicle's relevant surrounding area. However, the proposed solution solves only the active perception problem specific to its respective use case.

Despite the existence of some work exploring the concept of active perception, there are almost none that addresses the problem of integrating active perception within agents reasoning cycle in a more general way. One exception is [6] where the authors proposed a solution to integrate active perception at the architecture level of a BDI agent. The architecture was modified in order to apply active perception plans to reveal missing beliefs, unknown or outdated, that are used as preconditions for plans. The authors [6] proposed four algorithms: IAP, ITAP, SAP, and DSAP. The first one, IAP, reveals all the missing beliefs of the agent, thereby, it guarantees that the optimal plan is selected. However, since it is likely to perform unnecessary active perception plans its performance is not ideal, except when executing active perception plans that have no cost. The second one, ITAP, allows the agent to choose between performing an active perception plan or a feasible plan, then if perception plans have cost it may choose, at any time, to perform a feasible plan instead of performing all the perception plans. With this, the algorithm is able to reduce the cost of the active perception, but it is not guaranteed that the optimal plan will be selected. The third one, SAP, demands that the agent commits to a plan before performing the active perception plans that reveals it, and then if it becomes feasible that the agent can choose to execute it or select another plan to reveal. Lastly, DSAP, also demands the agent to commit to a plan to be revealed and additionally allows the selection of the order of execution of the active perception plans that reveal it. These algorithms seem to be a suitable approach to solve the problem of integrating active perception with the BDI architecture. However, the paper lacks information related with how the active perception mechanism was developed and properly integrated within the BDI reasoning cycle. Besides, there is no implementation available or experimental results showing its effectiveness.

3 ACTIVE PERCEPTION MODEL

This section describes the model of active perception that is being proposed in this work. It starts with some definitions and is followed by a modified BDI reasoning cycle that considers active perception.

3.1 Definitions

Beliefs are the representation of the information that the agent has about the world or itself. They can be acquired via perception, messages from other agents, or reasoning. In addition to this definition, beliefs can be further distinguished into: **timed beliefs**, their difference is that they have a lifetime, when they are not updated for longer than its lifetime they are considered outdated and, therefore, it can no longer be trusted; **active perception beliefs** (APB) are subject of active perception; APBs can be further differentiated in **timed active perception beliefs** (TAPB), which are subject to active perception and have a lifetime; and **missing beliefs** are outdated beliefs or those not included in the agent's belief base yet;

Desires are the states of the world the agent wants to achieve but is not committed to yet. A particular type is **active perception desires** (APD): the desire to change the state of a missing belief into an updated belief.

Intentions are desires the agent is committed to accomplish. A particular type is **active perception intentions** (API): the intention to change the state of a missing belief into an updated belief.

Plans represent a behaviour strategy (e.g. a sequence of actions) to achieve some intention. A particular type is **active perception plans** (APP): a plan to achieve an API. **Revealing** is the process of executing an APP to change the state of missing beliefs into updated beliefs [6].

Active perception selection pressure Two important aspects when designing the active perception mechanism are the resulting frequency at which APPs are performed and the credibility of each active perception belief. The former is important because applying APP constantly results in more computational resources being used, and less time being spent in performing regular plans. In real world scenarios this is aggravated, since performing an APP implies in consuming resources.

The credibility of APB is the probability of an APB being correct, and it is directly related to the frequency at which APPs are performed and the dynamism of the environment. Suppose that an agent has the APB that it is daytime and it has no information about its location and current season, if it executes an APP to verify if it is daytime at 1:00 pm and then only performs an APP again at 7:00 pm, it could not be sure if it was still daytime at 6:30 pm, since the sunset depends on the region of the world and the current season, therefore, in this situation the credibility of this APB would be low. However, if an APP is performed within an interval of 5 minutes or if the agent knew in which region of the world it is located and

what is the current season, it would be more sure if it is daytime at 6:30 pm, thus, in this new scenario the credibility of this active perception belief would be high.

It is important to find the right balance between the active perception plans frequency and active perception beliefs credibility. However, there is no right answer for this trade off, since it is directly tied to the dynamism of the environment, and each specific use case. For future reference this trade off will be called active perception selection pressure, when the APP frequency and the APB credibility is high the active perception selection pressure is also high, and when they are both low the active perception selection pressure is also low.

3.2 Modified reasoning cycle

The traditional BDI architecture proposed by Wooldridge [12] consider that the agent has all the necessary beliefs required to deliberate which intentions it should commit to achieve. However, especially in real world scenarios, this assumption is not reasonable since beliefs may be unknown or outdated, and if this is not taken into account, the agent deliberation process may lead to the selection of suboptimal actions. One approach for solving this problem is to drop the assumption that all beliefs are known and updated, and based on the agent internal state (beliefs, desires, and intentions) try to actively perceive what is required for the deliberation process.

As an example of the process of active perception, lets analyze a scenario of a search and rescue mission using unmanned aerial vehicles (UAVs). In this context, the agent is an UAV that carries a buoy and a camera facing down. Suppose that the agent receives a message informing the location of several drowning victims, when this happens it immediately takes off to deliver a buoy to the nearest victim. However, when it reaches the location of the first victim it does not assume that the belief of the victim's location is always known and updated, in other words, consider that the victim's location is an timed active perception belief. This is because the victim may have already drowned, moved or has already been rescued and, in these cases, it would not be wise to drop the buoy for a victim that is not there, if this happens the agent should fly to the next nearest victim and try to rescue it instead. Thus, before dropping the buoy, the agent actively tries to perceive if the victim is in the informed position.

With the process of practical reasoning proposed by Wooldridge [12] as the starting point, the active perception can be included as an intermediary process between the *options generating function* and *filter function*, this can be seen in listing 1 (lines 6–10 are included for active perception).

Listing 1: BDI reasoning cycle with Active Perception

```
function action(p:P):A
begin
    B = brf(B,p)      # belief revision function
    D = options(B,I) # possible desires

    APD = optionsAP(D, APB)
    if(has APD)
        API = filterAP(APB, APD, I)
        APB = execute(API)
        return action(perceive())
    else
        I = filter(B,D,I) # commitment to a desire
        return execute(I) # actions to achieve I
end
```

Based on the new desires and active perception beliefs (APB), an *active perception options generating function* (optionsAP) is applied to verify if any of the APB that are related to the agents desires are missing beliefs, if there are any missing beliefs the agent acquire an active perception desire (APD) to reveal them. In the context of the UAV example, lets assume that the belief of the victim's location has a lifetime of 1 minute, in case the agent reach the victim's position after 1 minute or more of receiving the information, that belief can be considered a missing belief, thus it would generate an APD to verify if the victim is in the indicated position.

If there are any APD, an *active perception filter function* (filterAP) generates an active perception intention (API) to reveal the missing beliefs. In the example of the UAV, since there is only one APD, the filterAP function would commit to the desire of verifying if the victim is in the indicated location or in a small area around it.

Lastly, based only on the API, the *execute function* select the actions that tries to reveal the missing beliefs. For the UAV, this would represent turning on the camera, running recognition algorithms, tracing a flight path, and actually flying. In the case where AP is triggered, the algorithm starts again (line 10) to perceive the new environment and properly decide what to do based on that.

As pointed out, our proposal maintains all the steps of the traditional BDI reasoning cycle, resulting on a model that contains a mix of passive and active perception. The advantages of both approaches are leveraged, and the disadvantages of each method are mitigated by the other.

4 DESIGN OPTIONS

The concrete implementation of the proposed abstract model can be done in several different ways. This section discusses some of these alternatives.

4.1 Plans representation

Plans play a major role when programming BDI agents since they contain instructions to fulfill an intention. Moreover, they are intrinsically related to the active perception mechanism. In this work, plans are composed of a name, an associated desire it is a candidate to fulfill, a sequence of instructions (plan body), and the context where it is suitable. A single desire may have multiple different plans (called relevant plans), those that are suitable in a particular context are called applicable plans. When a particular plan is chosen for execution, we say that the agent is committed to the corresponding desire, and the agent now intends it. In this section, plans are written as follows to help us to exemplify the proposal and its alternatives:

```
<plan name> for <desire>:
     preconditions = <belief> and <belief> and ...
     plan body     = <action> and <action> and ...
```

4.2 Distinction of Beliefs

The first aspect that is taken into account is how to differentiate regular beliefs from active perception beliefs (APB). Two approaches are considered: annotated plans and annotated beliefs. The annotated plans approach **(1)** consists of marking plans as requiring active perception and considering all beliefs in their preconditions as APB. Therefore, before selecting a plan for execution, it must be checked whether the precondition beliefs are missing beliefs, for those that are considered as missing, an APP must be carried out

if one is available, and if not, it is treated as a regular belief. The following plan illustrates this case, where `plan1` is marked with `[ap]` indicating that `belief1` and `belief2` are both APB.

```
plan1[ap] for desire1:
    preconditions = belief1 and belief2
    plan body     = action1 and action2
```

The annotated beliefs approach **(2)** consists in marking, individually, the beliefs in the preconditions as being APB, thus for the marked ones an active perception plan is performed if they can be considered missing. This alternative is illustrated below, where only `belief1` is marked with `[ap]`, indicating that just this belief is subjected to active perception.

```
plan1 for desire1:
    preconditions = belief1[ap] and belief2
    plan body     = action1 and action2
```

Among these two alternatives, marking of beliefs (2) seems to be a more natural and practical approach since the necessity of active perception is based on the characteristics of beliefs, and not plans. Besides, when marking the whole plan (1) the ability to distinguish regular beliefs and active perception beliefs is lost.

4.3 Analysis of relevant plans

Another designing choice that has to be made is **(1)** if all the active perception plans referring to all the available relevant plans should be performed beforehand, and only then check if the preconditions are satisfied (grouped mode), or **(2)** to start by checking the context of the relevant plans one by one and only perform active perception when it is needed (individual mode).

Taking listing 2 as an example, suppose the agent has the *desire1* which has two relevant plans: *plan1* and *plan2*. The grouped mode approach consists to perform the active perception plans for *belief1*, *belief3*, and *belief4* and only then verify if the contexts *(belief1[ap] and belief2)* and *(belief3[ap] and belief4[ap])* are satisfied. The individual mode consists to first execute the active perception plan just for *belief1* and check if the context *(belief1[ap] and belief2)* is fulfilled and only if it fails proceed to perform the active perceptions plans for *belief3* and *belief4*.

Applying active perception for all relevant plans grouped is similar to the algorithm IAP proposed in [6], where the authors concluded that IAP guarantees that the best plan is selected, but it is only optimal when all the missing beliefs must be revealed in order to choose the most advantageous plan, or when the active perception plans have no cost. This conclusion has the assumption that the BDI reasoning cycle algorithm for selecting a relevant plan to commit guarantees that the best plan is selected, however, it is not uncommon for BDI architectures to have strategies that can not assure that the best plan is selected. Thereby, the grouped mode can only be considered the best choice when all these conditions are met, otherwise it may perform unnecessary active perception plans.

Listing 2: Plans example

```
plan1 for desire1:
    preconditions = belief1[ap] and belief2
    plan body     = action1 and action2

plan2 for desire1:
    preconditions = belief3[ap] and belief4[ap]
    plan body     = action3 and action4
```

4.4 Context verification

Another important aspect to consider is the context verification, this can be done with two distinct methods. The first method **(1)** is to reveal all the missing beliefs and only then check whether the context is true or false. Using listing 2 as an example, in *plan2* the beliefs *belief3* and *belief4* would be revealed and only then the whole context would be checked, in the case that *belief3* is false it is unnecessary to perform an active perception plan for *belief4*.

This problem is mitigated with the second method **(2)**, in the case the relevant plans are being analyzed in the individual mode, which consists in verifying one-by-one each missing belief after they are revealed whether the context can still be true. With listing 2 as an example, when *belief3* is revealed it is verified if it is true or false and if it is false no AP plans would be performed for *belief4*.

Thus, method (2) is advantageous because it might avoid performing unnecessary active perception plans. However, the process of verifying the context more frequently could increase the computational complexity or the required processing power.

4.5 Timed active perception belief lifetime

An important characteristic of the proposed model is the possibility of a belief being outdated. It is thus essential to consider how a belief would change its state from updated to outdated.

A first method **(1)** to out-date beliefs is to add a mark into the plan or belief to indicate for how long the belief should be considered updated once revealed. In the case of annotated plans, such as in listing 3, the whole context should be revealed and verified within the time limit, in this case $1000ms$. For annotated beliefs, like in listing 4, each belief has a lifetime, i.e., how long it is updated after revealed. In the example, *belief1* is considered outdated after $1000ms$ and *belief2* is considered outdated after $3000ms$.

Some variations can be considered for the notion of being outdated. Another method **(2)** is to consider beliefs updated from the revealing moment until a plan is selected, once used (to select a plan) it becomes outdated. In the case that the relevant plans are being checked in the individual mode (section 4.3), an alternative **(3)** is to consider beliefs as updated while the context is being verified.

All three alternatives seem reasonable, but the period a belief is updated impacts on the active perception selection pressure and consequently the AP belief credibility.

Listing 3: Annotated plans with lifetime

```
plan1[ap, 1000] for desire1:
    preconditions = belief1 and belief2
    plan body     = action1 and action2
```

Listing 4: Annotated beliefs with lifetime

```
plan1 for desire1:
    preconditions = belief1[ap,1000] and belief2[ap,3000]
    plan body     = action1 and action2
```

4.6 Revealing order

Another aspect to take into account is the execution order of the active perception plans. A possibility **(1)** is to follow the order in which the active perception beliefs appear, another method **(2)** is to perform the active perception plans for the AP beliefs with greater lifetime first.

Using listing 4 as example, for the first approach the active perception plan for *belief1* would be performed before the AP plan for *belief2*. For the second method, since *belief2* has a greater lifetime it would be revealed before *belief1*.

The second method may increase the chances of the context being valid until all beliefs are revealed. Other than this, there is no clear advantages or disadvantages between both methods.

4.7 Selected Options

The design options we have so far are: three different approaches for defining the active perception beliefs lifetime, two distinct ways to indicate beliefs subjected to AP, two analysis of relevant plans, two context verification, and two revealing orders. If all the options were combined it results in 48 possibilities. However, there are two combination of options that can not be together: revealing the relevant plans in grouped mode with beliefs lifetime being valid until the end of context, and relevant plans in grouped mode with context verification being done one by one. Hence, there are 28 valid combinations of design options, therefore, it would be exhaustive to analyze all possible combinations. Thus, considering the advantages and disadvantages of each design option presented in the previous sections, Table 1 contains the combinations that look most promising.

Approach 1 uses annotated beliefs, it reveals all the relevant plans beforehand, it only checks the context after all the revealing is completed, the beliefs do not expire until all the relevant plans are analyzed, and the active perception plans are applied in the order they appear. This makes this approach the one with less active perception selection pressure out of the five, since all AP beliefs are revealed beforehand and they last until all relevant plans context are analyzed or one is selected. As a result, active perception plans are performed less frequently, all of them are executed exactly once at the beginning of the process. This method is advantageous when all active perception beliefs would need to be revealed anyway, otherwise, it would execute unnecessary AP plans, which can be really costly in real world scenarios, and when the environment is not very dynamic, since its AP beliefs credibility is low.

Approach 5 uses annotated beliefs, it reveals the relevant plans individually, the context is verified after revealing each missing belief, the beliefs expire with time, and the active perception plans are applied for the AP beliefs with greater lifetime first. This results in this approach being the one with more active perception selection pressure out of the five. With this, active perception plans are performed more frequently, since active perception beliefs can expire between checking the relevant plans. This method is advantageous when it is not required that all AP plans are executed. And it proves to be advantageous in more dynamic environments, since its AP beliefs credibility is high.

Approaches 2, 3, and 4 are in between 1 and 5 in relation to the active perception selection pressure. In any case, it is difficult to assess objectively which method is better, since each approach can prove to be more efficient in different situations. It is possible to notice that designing an active perception mechanism does not have an obvious answer and it is not a trivial task.

5 IMPLEMENTATION

The proposed model for active perception is implemented and integrated with Jason programming language [3] in order to evaluate the proposal in practical scenarios.

The implementation of our 5 selected approaches by re-implementing the Jason interpreter would involve a lot of steps and it would not be a trivial task. Since it is not yet clear that the proposed model is satisfactory and which of the approaches presented in section 4 is the better, we opted for a simpler and faster implementation method based on plan transformation. Once different approaches are tested and evaluated, it would be interesting to use the Jason architecture modification as a definitive solution.

The proposed transformation algorithm can be seen in listing 5. In summary, before executing the agent logic, the syntactic substitution algorithm checks which plans have active perception beliefs as precondition and, based on that, new plans are created using Jason *directives* [4]: "directives are used to pass some instructions to the interpreter that are not related to the language semantics, but are merely syntactical."

Listing 5: Syntactic substitution algorithm

```
procedure SyntaticSubstitution(PlanLibrary):

  let PlansWithAp be a list

  for all Plans in PlanLibrary:
    if Plan has ap belief in context:
      PlansWithAp.add(Plan)

  PlanLibrary=Directive(PlanLibrary,PlansWithAp)
  return PlanLibrary
```

The application of the syntactic substitution algorithm in the Jason program represented in listing 6 would identify that one of the relevant plans for the desire *!g* contains active perception beliefs as precondition. We have different transformations for each approach discussed in the last section. We also assume that the original program has APP for APB as shown in listing 7.

Listing 6: Jason program before syntactical changes

```
!g.
+!g: b[ap(1000)] & d[ap(3000)] <- .print("GOAL1").
+!g <- .print("GOAL2").
```

Listing 7: Active perception plans

```
+?b[ap] <- ... // some code to reveal b
+?d[ap] <- ... // some code to reveal d
```

By applying the syntactical substitution algorithm with Directive2 (approach 2 in Table 1) in the Jason program shown in listing 6, we have the program of listing 8. The triggers *+!g* are replaced by *+!g[rp]*. A new plan with trigger *+!g[ap]* is added, it calls the plan *update* for each belief marked as requiring active perception, and then sets up the intention *!g*. The *update* plan is added and it is responsible for initiating the active perception plan, this takes place by triggering an event like *+?b[ap]* for the belief used as argument. However, this only occurs when the belief is missing. To check whether a belief is outdated, the internal action *ap.updated* is used, it is *true* when the AP belief is updated and *false* otherwise.

When applying the syntactic algorithm with the Directive3 in the program illustrated in listing 6, it results in listing 9. The event *!g* is replaced by *!g[ap, l_1]*, the plans *+!g[ap, l_x]* are added in

Table 1: Active perception mechanism options

Approach	Annotation	Mode	Context Verification	Belief Expiration	Revealing Order	AP Selection Pressure
1	Beliefs (2)	Grouped (1)	All (1)	Relevant Plans (2)	Natural (1)	+
2	Beliefs (2)	Grouped (1)	All (1)	Time (1)	Natural (1)	++
3	Beliefs (2)	Individual (2)	All (1)	Time (1)	Natural (1)	++
4	Beliefs (2)	Individual (2)	One by One (2)	End of Context (3)	Greater lifetime (2)	++
5	Beliefs (2)	Individual (2)	One by One (2)	Time (1)	Greater lifetime (2)	+++

Listing 8: Jason program after syntactical changes (Approach 2)

```
!g[ap].

+!g[ap]
    <-  !update(b[ap(1000)])[ap];
        !update(d[ap(3000)])[ap];
        !g[rp].

+!g[rp]: b[ap(1000)] & d[ap(3000)] <- .print("GOAL1").

+!g[rp] <- .print("GOAL2").

+!update(X[ap(T)])[ap]: not ap.updated(X,T) <- ?X[ap].
+!update(_)[ap].
```

order to call the *update* plan for the active perception beliefs used as precondition of the respective *+!g*, the plan's triggers *+!g* are changed to *+!g[rp, l_x]*, with *x* being the number of the plan, and an additional *+!g[rp, l_x]* is added for each existing *+!g[rp, l_x]* in order to call *+!g[rp, l_x+1]*. The *update* plan is the same used by Directive2.

Listing 9: Jason program after syntactical changes (Approach 3)

```
!g[ap, l_1].

+!g[ap, l_1]
    <-  !update(b[ap(1000)])[ap];
        !update(d[ap(3000)])[ap];
        !g[rp, l_1].

+!g[ap, l_2] <- !g[rp, l_2].

+!g[rp, l_1]:b[ap(1000)] & d[ap(3000)] <-.print("GOAL1").

+!g[rp, l_1] <- !g[ap, l_2].

+!g[rp, l_2] <- .print("GOAL2").

+!update(X[ap(T)])[ap]: not ap.updated(X,T) <- ?X[ap].
+!update(_)[ap].
```

By applying syntactic substitution in Jason it is possible to imitate the behavior of the proposed reasoning cycle to include active perception. This approach advantage is that it does not require that all the details of Jason's reasoning cycle implementation are known to experiment the alternatives of active perception. However, since syntactic substitution is not exactly equivalent to modifying the reasoning cycle, the active perception mechanism is not guaranteed to work correctly for all situations.

All the implementations related to the active perception model discussed in this chapter can be found at ommittedbyblindreview.

6 EXPERIMENTS

To evaluate the proposed active perception mechanism, a MAS composed of UAVs is used to simulate a search and rescue scenario for drowning victims, similar to the example described in section 3.2. Three experiments are performed. The first consists of adding active perception to the action of dropping a buoy. In the second experiment we add active perception to the communication between UAVs. The third is a combination of the other two. The code of the experiments and more details to reproduce it can be found at x.

In the experiments, Gazebo is used to simulate a virtual world with victims, where the scouts cover an area looking for victims in a boustrophedon path, and the rescuers drop buoys for the victims (figure 1). PX4 is used as flight controller emulator. Mavros is used to interface PX4 with ROS, and the Jason-ROS package [8] is used to integrate Jason agents with ROS.

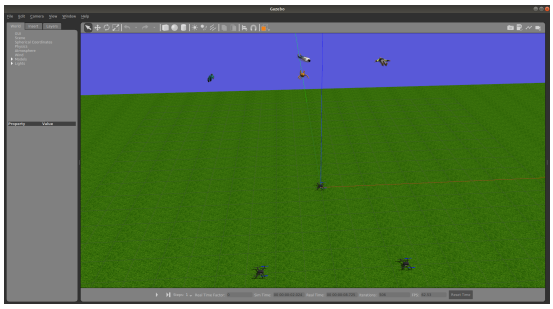

Figure 1: Gazebo search-and-rescue world

6.1 First scenario

The first scenario is almost the same as the search-and-rescue mission described in section 3.2. The difference is that victims are guaranteed to drown after a predetermined time.

Active perception is used in the plan to drop buoys, in the plan precondition we have a timed active perception belief (TAPB) of the victim's position. Before dropping a buoy, the agent checks whether the belief of the victims position is a missing belief and, if it is, an active perception plan is executed to reveal it. In this scenario, the AP plan consists in turning the camera on and checking whether the victim is still in the informed position.

To understand how the dynamism of the environment influences the importance of active perception, the missions are performed for the two different victims drowning times as shown in table 2. The drowning time of the first configuration were chosen in order to leave the environment dynamic enough for some victims to drown during the execution of the mission, while the times in the second

configuration were chosen so that the mission is completed before any victims drown. The first configuration is thus more dynamic and the second less dynamic.

Table 2: Victims drowning times

Setup	Drowning Time [ms]					
	Victim 1	Victim 2	Victim 3	Victim 4	Victim 5	Victim 6
1	70000	45000	48000	80000	60000	85000
2	100000	100000	100000	100000	100000	100000

To assess the results of the missions, the main aspect taken into account is the number of victims rescued and, subsequently, the efficiency of the system, which is calculated by dividing the number of victims rescued by the number of buoys dropped. Since the simulation is non deterministic, the results may vary for each run, thus each experiment setup is repeated five times and the statistical results are presented.

For the active perception mission, two different values are used for the lifetime of the timed active perception belief (TAPB): a shorter and a longer time, 5000 and 25000ms, respectively. To measure the impacts of using AP, the missions are repeated with and without AP and their results are compared. Also, it is performed for both AP approaches from table 1, which the implementations are described in section 5. The results obtained with the respective configurations are presented in table 3.

Table 3: Scenario 1 results

Experiment	Setup	AP	TAPB lifetime [ms]	Victims rescued			Buoys dropped			Efficiency w/ mode
				Mean	Std	Mode	Mean	Std	Mode	
1	1	NO	-	2.80	0.45	3	6.00	0.00	6	0.5
2	1	Approach 3	25000	2.80	0.45	3	4.00	0.71	4	0.75
3	1	Approach 3	5000	3.80	0.45	4	4.00	0.71	4	1
4	1	Approach 2	5000	3.60	0.89	4	3.80	0.45	4	1
5	2	NO	-	6.00	0.00	6	6.00	0.00	6	1
6	2	Approach 3	5000	6.00	0.00	6	6.00	0.00	6	1

For the first victims setup, experiments 1 to 4, it is possible to notice that experiments 1 and 2 had a worse result than 3 and 4, with most of the time 3 and 4 victims being rescued, respectively. In the second victims setup, experiments 5 and 6, there are no difference in the number of victims rescued between the experiment with active perception and the one with only passive perception.

In experiment 1, UAVs do not perform active perception before dropping the buoy. The rescuers sometimes drop a buoy for victims who have already drowned and, when a buoy is dropped, the UAV must return to its initial position and land to load a new buoy. This consumes time that could be used to rescue a victim who has not yet drowned.

Experiment 2 has a similar problem, since the TAPB lifetime is long, the AP pressure is not enough and sometimes the rescuers do not perform AP when necessary. Again, time is wasted dropping buoys for victims who have already drowned.

In experiment 3, the TAPB lifetime is shorter and consequently the AP pressure is higher. AP thus is always carried out before throwing a buoy which ensures that no buoy will be thrown in vain, increasing the efficiency of the mission. In this specific case, constantly performing active perception plans does not affect the agent's performance, since the AP plan has almost no cost, it consists only of turning the camera on and off and waiting 1 second.

For experiment 4, AP is replaced from approach 3 to approach 2, which has no impact on the mission result. This is expected, since the agent has only one relevant plan to drop the buoy that has a TAPB as a precondition, with the only difference between approaches 2 and 3 being how the relevant plans are revealed.

In experiments 5 and 6 the environment does not change fast enough for AP to have any impact on the mission, and since performing the AP plan has almost no cost it does not bring any disadvantages for the mission.

6.2 Second scenario

The second scenario is a modification of the first scenario, now the victims do not drown and active perception is not applied before dropping buoys, and instead of assuming that the network connection between the agents is constant and without losses the connection is limited to a predefined distance.

To consider AP in this scenario, a TAPB indicating if the UAV is in communication range is added in plans that send messages. Before sending messages the agent checks if the communication range belief is a missing belief and, if it is, the agent performs an AP plan to reveal it. The AP plan consists of flying to a predefined position where the UAV has communication range with most of the mission area. The communication range is limited to 10m and the communication range TAPB lifetime is 1000ms in the scout agent. The AP mission is repeated for different TAPB lifetimes in the rescuers agents, and for both active perception approaches. The setups and results of each experiment can be seen in table 4.

Table 4: Scenario 2 results

Experiment	AP	Rescuer range lifetime [ms]	Victims rescued			Buoys dropped			Efficiency w/ mode
			Mean	Std	Mode	Mean	Std	Mode	
1	NO	-	1.80	0.45	2	1.80	0.45	2	1.00
2	Approach 2	5000	6.00	0.00	6	6.40	0.55	6	1.00
3	Approach 3	5000	6.00	0.00	6	6.60	0.89	6	1.00
4	Approach 3	1000	6.00	0.00	6	6.20	0.45	6	1.00
5	Approach 3	10000	6.00	0.00	6	7.40	1.14	7	0.86

Experiment 1 has no AP. We can observe that with this configuration most of the time only 2 victims are rescued (mode), which is the smallest number among the 5 experiments. This happens because when the scout sends the location of victims 3 to 6 it is already out of the communication range. It can also be noted that the efficiency calculated with mode 1.00, which means that when the mission is successfully performed and no redundant buoy is dropped.

For all the experiments performed using active perception, 2 to 5, all the victims were rescued in all executions, since the mean is 6 and the standard deviations is 0. The only difference is regarding the efficiency of the system, in experiment 5 most of the time one unnecessary buoy is dropped, this is due to the larger TAPB lifetime which leads to an insufficient active perception pressure. Consequently, the rescuers sometimes do not perform active perception before communicating when it would be necessary, failing to inform that a victim was already rescued by them.

An interesting aspect to be highlighted in this scenario is that, although active perception has a cost related to flying to predefined positions, executing AP constantly does not bring disadvantages to the mission, as the environment is static.

6.3 Third scenario

The third scenario is a combination of the last two, the victims drown after a predefined amount of time and the network connection is limited to a certain distance.

The AP is included in the same way as the last scenarios, but in this case there are two TAPB instead of only one. The communication range is limited to 10m, and for the AP missions the victim's position and the scout communication range TAPB lifetime are set to 5000ms and 1000ms, respectively. The AP approach, the victims drowning times, and the rescuer communication range TAPB lifetime are varied to verify their impact on the mission results. The experiments configurations and results are presented in table 5.

Table 5: Scenario 3 results

Experiment	Setup	AP	Rescuer range lifetime [ms]	Victims rescued			Buoys dropped			Efficiency w/ mode
				Mean	Std	Mode	Mean	Std	Mode	
1	1	NO	-	1.60	0.55	2	2.00	0.00	2	1.00
2	1	Approach 2	5000	1.20	0.45	1	1.20	0.45	1	1.00
3	1	Approach 3	5000	1.20	0.45	1	1.20	0.45	1	1.00
4	1	Approach 3	10000	1.60	0.55	2	1.60	0.55	2	1.00
5	1	Approach 3	1000	1.00	0.00	1	1.00	0.00	1	1.00
6	2	NO	-	2.00	0.00	2	2.00	0.00	2	1.00
7	2	Approach 3	5000	4.40	0.55	4	5.20	0.45	5	0.80
8	2	Approach 3	10000	4.20	0.45	4	4.60	0.55	5	0.80
9	2	Approach 3	1000	3.60	0.55	4	4.20	0.45	4	1.00

Experiment 1 and 6 have only passive perception and their behavior are similar to the first experiment in scenario 2. The difference is that in experiment 1, in some executions, the victims drown before being rescued, however, in most cases, 2 victims are rescued.

For the first victim's setup, the AP experiments 2, 3, and 5 have worse results, only experiment 4 has the same performance as the experiment with only passive perception. This happens because the AP plans for the communication range TAPB take a considerable amount of time to be performed, so before they have time to complete, the victims drown. Thus, it can be concluded that the AP pressure for experiments 2, 3, and 5 is too high for an environment with this amount of dynamism.

For the second victim's setup, most of the time the AP experiments 7, 8, and 9 have the same result, 4 victims rescued, which is better than experiment 6, 2 victims rescued. However, experiment 7 has a slightly higher average of victims rescued, which is an indicative that it is better suited for this victim's setup. It is interesting to notice that experiment 7 has an intermediary AP pressure in comparison with experiment 8 and 9.

After analyzing the results from both victim's setup, one can conclude that the amount of active perception pressure that can be applied in order to improve the mission results depends directly on the dynamism of the environment and the cost, e.g. time, of the active perception plans. The lower the dynamism of the environment, the less the need for active perception, as beliefs maintain their credibility for longer. In more dynamic environments, the credibility of beliefs diminishes quickly, which requires active perception to be carried out more often. The problem of performing actions more constantly is that when it has a medium/high cost it can impact negatively on the performance of the mission. As an example, in this scenario the mission has a time restriction to be completed and the active perception requires an amount of time to be completed, thus, if the AP is performed too often it can reduce the performance of the mission, so it can be more advantageous to have a lower credibility than to spend time performing AP.

The experiments performed in this section demonstrate that the proposed active perception model can be an asset in some application scenarios, such as in search and rescue applications.

7 CONCLUSIONS AND FUTURE WORKS

This work concludes that it is not trivial to define an active perception model for BDI agents and to transform such model into a concrete implementation. It requires that several concepts related with BDI agents are expanded and several new ones created. In addition, it shows that it can be done in several different ways, leaving countless possibilities to be explored.

With the experiments performed, it was shown that it is possible to use the proposed model and implementations of the active perception mechanism with BDI agents programmed with Jason to create complex behaviours for mobile robots. And that active perception is important in some scenarios, where through its inclusion the efficiency of the system is improved.

As future works, one topic that can be further addressed is the concept of a belief being updated or outdated. In this work, a belief changes its state from updated to outdated when the time elapsed since the last update is longer than the belief lifetime, which is abrupt. Instead of having this binary change, it can be investigated the benefits of a a fuzzy transition, as the belief credibility decreasing over time.

Regarding the implementation of the AP mechanism, more approaches proposed in section 4.7 should be implemented and tested. After that, it should be analyzed the viability of modifying the Jason architecture to include the active perception mechanism as a definitive solution, and if it would bring any advantages.

REFERENCES

[1] Graeme Best, Oliver M Cliff, Timothy Patten, Ramgopal R Mettu, and Robert Fitch. 2018. Dec-MCTS: Decentralized planning for multi-robot active perception. *The International Journal of Robotics Research* (3 2018), 027836491875592. https://doi.org/10.1177/0278364918755924

[2] Graeme Best, Jan Faigl, and Robert Fitch. 2018. Online planning for multi-robot active perception with self-organising maps. *Autonomous Robots* (2018). https://doi.org/10.1007/s10514-017-9691-4

[3] Rafael H. Bordini and Jomi F. Hübner. 2006. BDI agent programming in AgentSpeak using Jason. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 3900 LNAI. Springer, Berlin, Heidelberg, 143–164. https://doi.org/10.1007/11750734{_}9

[4] Rafael H. Bordini, Jomi Fred Hubner, and Michael Wooldridge. 2007. *Programming Multi-Agent Systems in AgentSpeak using Jason.* John Wiley & Sons, Ltd, Chichester, UK. https://doi.org/10.1002/9780470061848

[5] U.G. Ketenci, Auberlet J.-M., Brémond R., and Grislin - Le Strugeon E. 2012. *Improved Road Crossing Behavior with Active Perception Approach.* Technical Report. http://perso.lcpc.fr/roland.bremond/documents/TRB12_Ketenci.pdf

[6] Niv Rafaeli and Gal A Kaminka. 2017. *Active Perception at the Architecture Level (Extended Abstract).* Technical Report. www.ifaamas.org

[7] Anand S Rao and Michael P Georgeff. 1995. *BDI Agents: From Theory to Practice.* Technical Report. www.aaai.org

[8] Gustavo R. Silva, Leandro B. Becker, and Jomi F. Hübner. [n.d.]. Embedded Architecture Composed of Cognitive Agents and ROS for Programming Intelligent Robots. In *Submitted to 1st IFAC-V World Congress.*

[9] Raymond So and Liz Sonenberg. 2009. The Roles of Active Perception in Intelligent Agent Systems. Springer, Berlin, Heidelberg, 139–152. https://doi.org/10.1007/978-3-642-03339-1{_}12

[10] Alois Unterholzner, Michael Himmelsbach, and Hans-Joachim Wuensche. 2012. Active perception for autonomous vehicles. In *2012 IEEE International Conference on Robotics and Automation.* IEEE, 1620–1627. https://doi.org/10.1109/ICRA.2012.6224879

[11] Danny Weyns, Elke Steegmans, and Tom Holvoet. 2004. Towards Active Perception In Situated Multi-Agent Systems. *Applied Artificial Intelligence* 18, 9-10 (10 2004), 867–883. https://doi.org/10.1080/08839510490509063

[12] Michael Wooldridge. 1999. Intelligent agents. *Multiagent systems* 35, 4 (1999), 51.