ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal
Regular Issue, Vol. 8 N. 2 (2019), 87-96
eISSN: 2255-2863
DOI: http://dx.doi.org/10.14201/ADCAIJ2019828796

Perception policies for intelligent virtual agents

Giancarlo Souza de Freitas[a], Thiago Ângelo Gelaim[a], Rodrigo Rodrigues Pires de Mello[a] and Ricardo Azambuja Silveira[a]

[a] University of Santa Catarina, Department of Informatics and Statistics, PPGCC, Campus Reitor João David Ferreira Lima s/n, 88040-900, Florianópolis, Brazil
giancarlo0295@hotmail.com, t.gelaim@posgrad.ufsc.br, rodrigo.mello@posgrad.ufsc.br, ricardo.silveira@ufsc.br

KEYWORD

Virtual Intelligent Agent; Perception; Sigon

ABSTRACT

Agents deployed to dynamic environments, such as virtual and augmented reality, need specific mechanisms to capture relevant features from the environment. These mechanisms enable agents to avoid process some useless information and act quickly. The primary goal of this work is to investigate the perception policies of an agent situated in a virtual environment. Perception policies allow giving more priority to sensors perceiving the changes occurring in the environment. Based on the proposed model, each sensor follows a strategy that can change its priority in the overall system. We developed two policies to change the sensors prioritization. The performance evaluation of the proposed model consists of comparing both approaches in a highly dynamic environment.

1. Introduction

Most of the abilities that an intelligent agent needs to interact with the environment in a virtual or a physical environment requires capturing, processing, and fusing information. Smart houses are examples of how many different ways human and software agents could interact with a cyber-physical environment. GPS data, wireless signal, cameras, and microphones are mixed to allow autonomous functions in a house, such as to improve Alzheimer patients' health care (Corchado et al., 2008).

The amount of data that can be available from different types of sensors in smart houses raises the need of taking into consideration the properties of dynamic environments. In such environments, the agents usually have to take actions at a short time. The agent may have a significant amount of perception, in which some of these do not represent new relevant information. Using an appropriate policy of using perceptions can reduce the agent's performance and decision-making processes time. Based on this, our primary goal is to propose perception policies and analyzes how they impact on the deliberation process of the agent over the agents' beliefs. The concept of perception policies helps the agent to perceive the environment correctly and prioritize the perception based on the application needs.

Several pieces of research found in the literature present strategies to model, filter and integrate perceptions and situation awareness in BDI agents (So and Sonenberg, 2009; Herrero et al., 2005; Farias et al., 2010; Van Oijen and Dignum, 2011; Stabile and Sichman, 2016; Rafaeli and Kaminka, 2017), In this work, we use a

practical approach to evaluate the role of perceptions in decision making. We model two strategies to represent the perception policies, where each policy presents a different approach to deal with the agents' sensors prioritization. Based on that, we add these strategies to a Sigon based agent reasoning cycle (Gelaim et al., 2019). Sigon agents model uses the Multi-Context Systems (MCS) approach that allows a loosening in its components. Sensors perception policies permit the agent to perceive the data in a dynamic environment in a more efficient way, where the sensors can prioritize some perception over others in accord with the environment properties.

The organization of this paper is as follows: Section 2 presents the context of the agent's perception. Section 3, presents the perception policies modeled in the Sigon agent. In Section 4 describes the evaluation of the policies and how they affect the agent's reasoning cycle. Section 5 presents the final remarks and future work.

2. Research Context

The recent evolution of tools designed to develop new Virtual Reality (VR) and Augmented Reality (AR) technologies allow (i) the evaluation of interaction models between agents and its surrounding, and (ii) the creation of new environment features. In the first case, creating real-life environments in VR enables to evaluate technologies such as smart houses and self-driving cars. This evaluation enhances how we can control and visualize the results of a model. To the second case, projections can add an environment interaction model. For example, in the perception of a fire risk, the system can project some notification to avoid this situation.

Perceptions are the primary mechanism in which an agent can update its knowledge about the environment. Gibson (Gibson, 1950) proposed that perception is the combination of the environment which the agent lives and how the agent acts in this environment. However, in a complex and dynamic environment, if the agent monitors all the aspects of the world, it may never decide what to do with all this knowledge.

The literature of perception and BDI-like agents mainly focus in active perception, i.e., the perceptions may change the agents' current goals (Weyns et al., 2004), or situation awareness, i.e. 'the perception of the elements in the environment within a specific time slice and space, the comprehension of their meaning, as well as the projection of their status in the near future' (Endsley, 1988). Based on these works, in our research, we focus on a practical, exploratory approach to the problem.

Bajcsy *et al.* (Bajcsy et al., 2018) argues that an active perceiver has the ability do decide why, what, how, when, and where to perceive. Based on these five components we focus on the 'when' to perceive. In this sense, we describe our approach as a step of active perception.

2.1. Sigon Agents

To manipulate multiple knowledge representations and a considerable amount of data the proposed model of an agent uses the multi-context system approach (Giunchiglia and Serafini, 1994). A multi-context system is composed of contexts and bridge rules. The first describes agent capabilities and the later the relationship between contexts (Brewka et al., 2011).

The Sigon framework, used in previous research (Mello et al., 2018) in the research group, is quite adequate to model the agent as a multi-context system. Gelaim *et al.* (Gelaim et al., 2019) define a Sigon agent as a multi-context system as follows:

Definition 1 *Let AG be an agent. AG is defined as:*

$$AG = \langle CC \bigcup_{i=0}^{n} C_i, \Delta br \rangle$$

where CC is the Communication Context and Ci with $1 \leq i \leq n$ are the contexts defined by the developer. A bridge rule bri is a rule connecting two or more contexts.

The agent interaction with the environment is an essential part of achieving its goals. Therefore, a Communication Context is an interface between the agent and its environment. Interaction consists of how the agent

perceives information and change the state of the world. This context consists of a set of sensors and actuators. A sensor is a sorted pair defined as:

$$S_i = (\omega, \chi), \tag{1}$$

Where ω is the sensor identification, and χ is the function that maps an observation to a perception. The actuator is also a sorted pair, and its definition is:

$$A_j = (\rho, \alpha), \tag{2}$$

Where ρ is the actuator identification, and α is the action to be executed. Based on that, the definition of the Communication Context is (**?**):

Definition 2 *Let CC be a Communication Context. The formal definition of CC is:*

$$CC = \langle \bigcup_{i=1}^{n} S_i, \bigcup_{j=1}^{m} A_j, \rangle$$

where S_i with $1 \leq i \leq n$ are the agents sensor, and A_j with $1 \leq j \leq m$ are the agents actuators.

The model of the belief, intention, and desire contexts are based on a logical context. A logical context in Sigon agent is based on Casali, Godo and Sierra (**?**) works, and is defined as follows:

Definition 3 *Let C be a logical context. C is defined as:*

$C = (L, Ax, \delta)$

Where L is the context language, Ax is the axioms set, and δ is the inference rules of the context. Logical contexts can be built using different types of logic, such as first order, propositional or dynamic logic (Gelaim et al., 2019).

In this research, we focus on how an agent can use perception policies to perceive the environment. Some of the perceptions may not represent new information and can decrease the agent's performance. Based on that, section 3 present policies in which the agent can prioritize and focus on perceptions that represent different and relevant information.

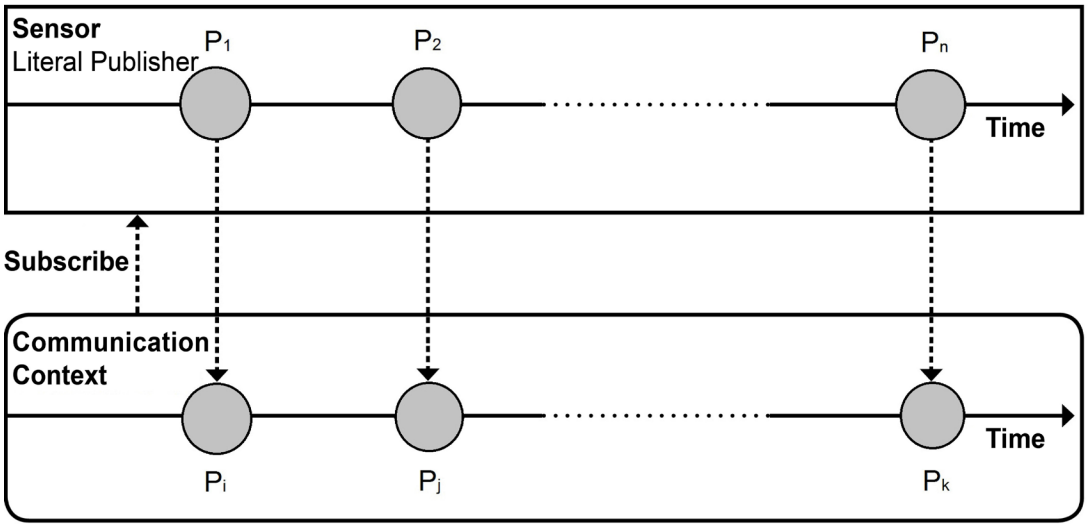

Figure 1: Representation of the sensors model. Adapted from (Gelaim et al., 2019).

3. Perceptions policies in Sigon Agents

In this section, we present the perception mechanism developed for deploying on Sigon agents. However, these policies are architecture independent. The primary goal of this approach is to enable the agent to perceive the dynamic aspects of a virtual environment more effectively.

As mentioned in section 2.1, the Sigon agents' Communication Context works as an interface between the agent and its environment. The agents' sensors perceive the current data from the environment. The sensors models have a literal publisher, in which they add the agents' perception into the Communication Context. Based on the language defined on Gelaim *et al.* (Gelaim et al., 2019),, the developer has the task to define the sensor and the literals publication. Figure 1 shows interaction between the sensors and the Communication Context.

sensor: SENSOR‘(*'identifier*‘,*' implementation*‘)*"*.*'*

he interface between Sigon agents and the environment was built using a client-server architecture. The communication occurs via sockets, enabling sensors to interact with the environment. Socket messages are the service employed for exchanging information. A message can contain information about the environment, such as an object and its coordinates. We use Horn Clauses to represent messages. For example, a message can represent a *cube* object at position *3.0, 4.0, 3.0* as:

$$message = position(cube, 3.0, 4.0, 3.0)$$

Assuming that this cube is moving, the sensor receives the new coordinates at the beginning of each reasoning cycle. Perceptions of this nature have a high rate of the update on the agent's sensors. For example, if the speed of the cube is 33,33m/s and the agent sensors' capture rate is 100fps. Therefore the agent has 100 significant perceptions each second. Now assuming the speed of the cube changed to 1m/s and the capture rate still is 100fps. In this case, the agent perceives the same position in different reasoning cycle. Based on these, we believe that a sensor should have its priority decreased as time passes, enabling the agent to have a higher focus on sensors that have a higher rate of usage.

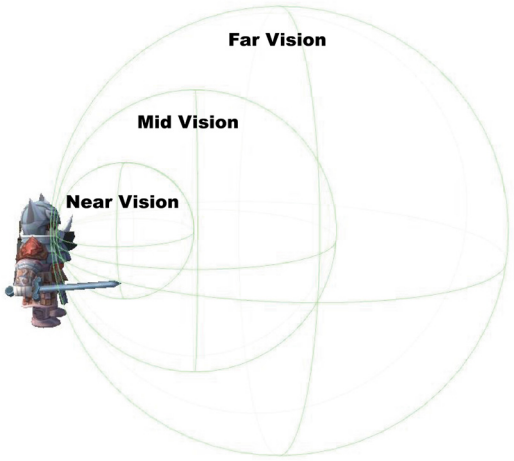

Figure 2: Agents' sensors vision example.

A sensor prioritization strategy must be defined to handle conditions, such as the previous example. Each sensor has a priority value in the interval [1,10]. The priority value can increase or reduce based on its update rate. For example, while the cubes' speed is 33,33m/s its priority is 10, and when the speed changes to 1m/s, its priority is decreased to 2. Equation 3 shows the agents' sensor encapsulation mechanism.

$$M = \sum_{i=1}^{n-1} S_{i-1} \subseteq S_i \tag{3}$$

Where M is the agents' perception mechanism. $Si-1$ is a sensor able to perceive a spectrum less or equals to Si. For example, the model organizes the vision of the agent in three sensors: near vision ($S0$), intermediate vision ($S1$), and far vision ($S2$). Each sensor represents a different level of distance between the agent and the object to be perceived. Figure 2 represents these example.

Based on the environment objects and in the agent sensors', different policies for priority strategy can be used. Subsection 3.1 presents a progressive perception policy. Subsection 3.2 presents a sudden perception policy.

3.1. Progressive perception policy

The progressive perception policy takes into consideration different perceptions received by a sensor. When a sensor stores a new perception, it checks the previous reasoning cycles perceptions and defines if the new perception represents different information. This policy states that a previous perception captured by the sensor should have a designated time to be stored. Progressive police enable the sensor to search for environment changes and process new data.

The development of the progressive perception police in Sigon agent uses two HashSets structures. Each HashSet stores a type of perception. *Cur* is a HashSet containing the current sensors' perceptions. Moreover, *Prev* is a HashSet of previous sensors' perceptions. After a new environment perception we define the following operations:

- The *Prev* HashSet receives the perception stored by the *Cur* HashSet. This resource enables the agent to verify whether this new perception represents some change in the environment. HashSet structures guarantee that the agent stores each element is just stored once. Taking this into consideration, the agent can determine if the *Prev* HashSet did not change, which means that the sensor did not store new perception since the last priority test;
- If the agent perceives that a new perception from the sensor It increases the priority is increased. Otherwise, the agent decreases it.

3.2. Sudden perception policy

The Sudden Perception policy focus on the amount of unforeseen perception detected in a specific moment. In this sense, it is a more reliable approach for dynamic environments. The agent can prioritize a particular sensor in a given moment. The definition of this policy is:

$$P = (1 - \frac{m}{F}) \cdot 10 \tag{4}$$

In which P is the sensor priority, F is the force added in the sensor, and m is the mass. In this case, F is the number of perception, and m is the configurable according to the environment dynamism. As the mass increases, the perception number tolerance decreases. Therefore the sensor keeps a low priority. On the other hand, a sensor receiving a high rate of perception must have its priority increased. If the P value is negative, it means that the number of perception is lower than the mass, hence the sensor priority value must be minimum. A constant negative P value implies in a more dynamic environment than the initial design. In this case, the mass must have a higher value.

Each sensor uses its own HashSet. The HashSet stores the number of perceptions until the next priority checking moment. When it occurs, the equation 4 is applied. After the priority update, the sensor removes the perceptions from the HashSet, allowing it to receive new perceptions.

Both strategies present different methods of changing sensors priority. The first policy gradually changes its priority based on new information. A possible scenario for this policy is a virtual game, in which sensors'

priority is less critical. In the second strategy, there is no guarantee of gradual change in the priority. Depending on the number of perceptions, it is possible for a priority goes from 5 to 10 at a time. This strategy could be used, for example, in health care environments, in which sudden changes may trigger an alert.

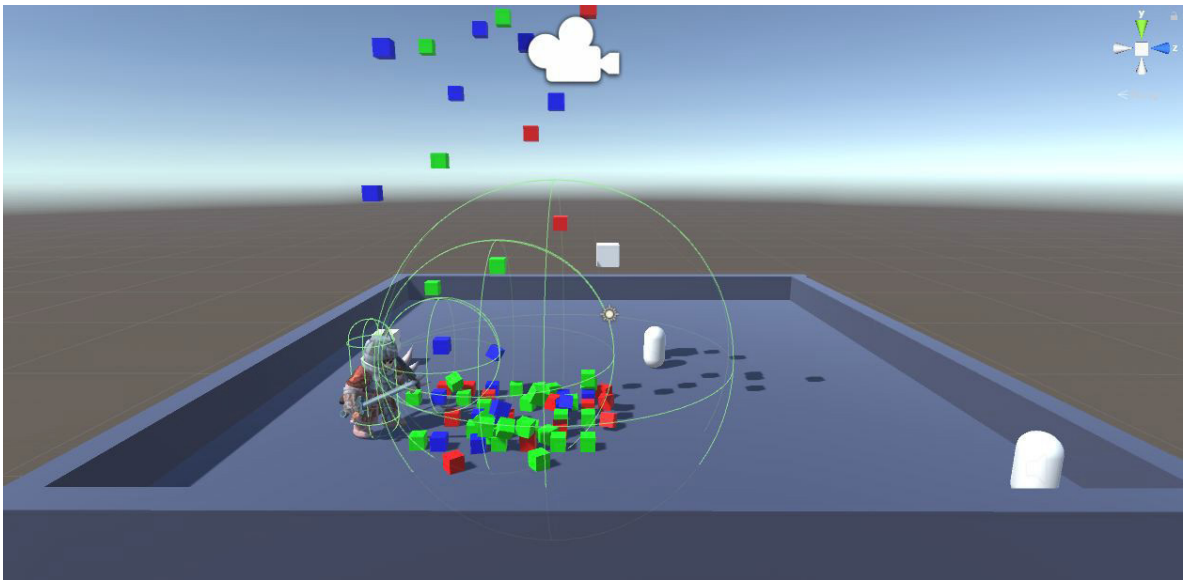

Figure 3: The agent and its sensors' acting in the environment.

4. Evaluation

To evaluate the proposed sensors policies, we used the Unity 3D Engine (Unity Technologies, 2019) to model and deploy a virtual scenario. The required communication process between the Sigon agent and Unity uses TCP sockets. The scenario consists of objects that should be perceived by the Sigon agent, such as walls, floor, static and dynamic blocks. A perception can determine the object current position. During execution, the dynamic blocks can appear in random positions of the Unity 3D environment. Using these types of blocks allows the agent to use the sensors prioritization policies and based on different blocks locations, verify whether it is a new perception. If a new perception represents a block in the same position, then the sensor can define it as an old one. Otherwise, the sensor defines the block as a new one. Figure 3 shows the scenario developed.

The first test performed in this Unity 3D environment evaluates the progressive perception policy. The publishing of perception into the agent's reasoning cycle is not part of this test goals. Sensors representing near, intermediate, and far vision were employed to collect data from the environment. The sensors' priority had different values at different periods. Figures 4 and reffig:perceptionsNumberGraphTest1 show the evolution of priorities and number of perceptions for these three sensors.

The environment used for the second experiment is the same as the first one. However, at this time, we consider the publishing of perception into the agent's reasoning cycle. This experiment shows that the time of the agents' reasoning cycle did not process all the environment changes. This limitation is probably resulting from using TCP protocol in the communication between the agent and the environment. TCP ensures that the agent receives all messages correct order. Therefore, it suspends the communication when a high amount of message is received and not computed. Figure 6 and 7 shows the elapsed time between the perception being sent from the environment and the agents' reasoning cycle over a perception. Since the sensor's operations are much simpler than the agent's reasoning cycle, a sensor can perceive more than 1000 times faster than the reasoning cycle. This fact states that unnecessary perceptions could affect the agent's performance. Therefore sensors' policies should be used.

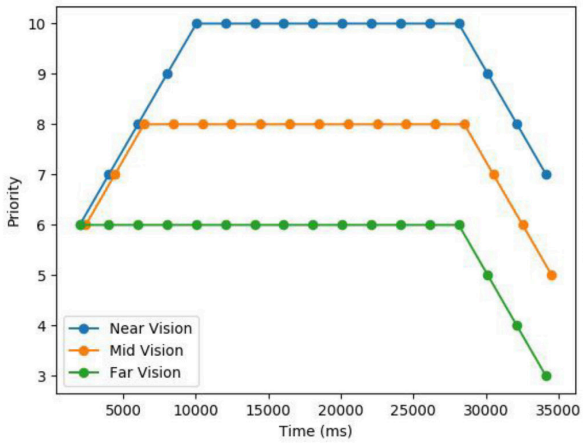

Figure 4: Experiment 1: sensors' priority.

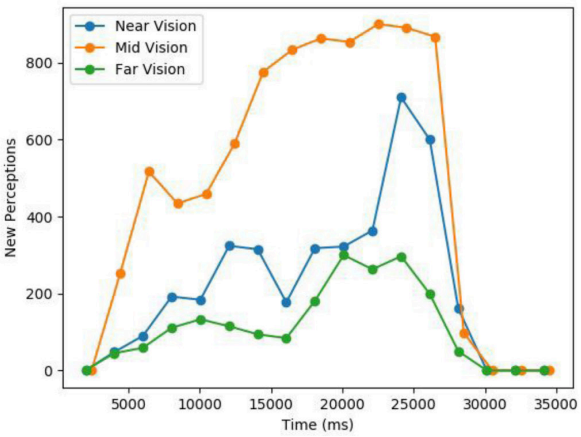

Figure 5: Experiment 1: sensors' number of new perceptions.

Based on these two experiments, it was defined the sent of perception should happen at every 50 milliseconds. This value is similar to the average time of the agent reasoning cycle. To avoid a TCP connection overload we set the time to a 0.5 milliseconds interval. This design decision allows processing of the maximum amount of perception during a cycle.

We also analyze the relationship between the perception of an object and its current state in the agents' knowledge in our experiments. The primary goal is to improve the agents' reasoning cycle and sensor integration. As an example, a Sigon agent stores data about a specific object, in which a sensor only publish a perception when it represents an environment change. This approach avoids storing unnecessary information, resulting in a higher tolerance process of the perception from the environment.

Subsequently, we perform a third experiment to evaluate the sudden perception policy. This experiment focuses on how this policy affects the agents' reasoning cycle time. It also answers the following research question: "What is the mass influence when the agents' number of perception is greater than the mass value?". To analyze this behavior, we established that each sensors' mass value equals to 50. A specific sensor also receives a higher amount of perception than other sensors. The sensor representing near vision was selected to receives more perceptions.

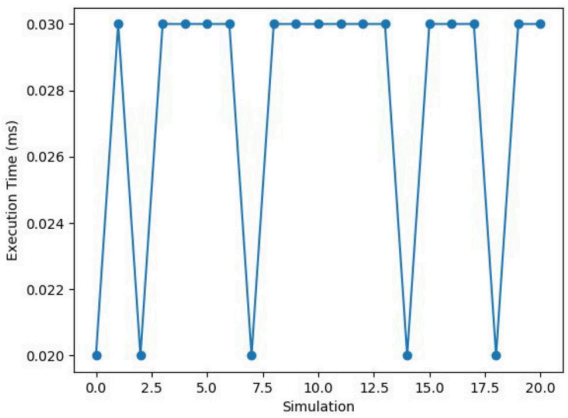

Figure 6: Experiment 2: perceptions sending time.

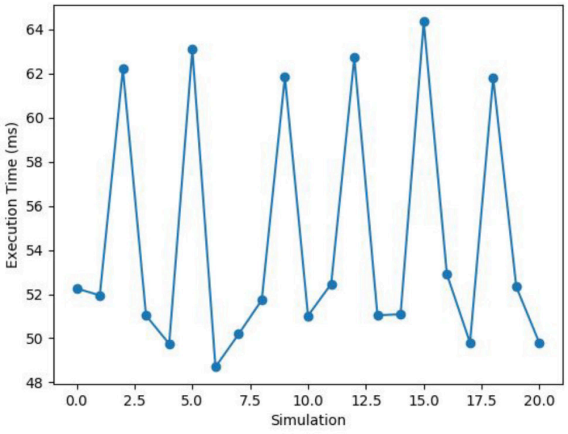

Figure 7: Experiment 2: agent's publishing time.

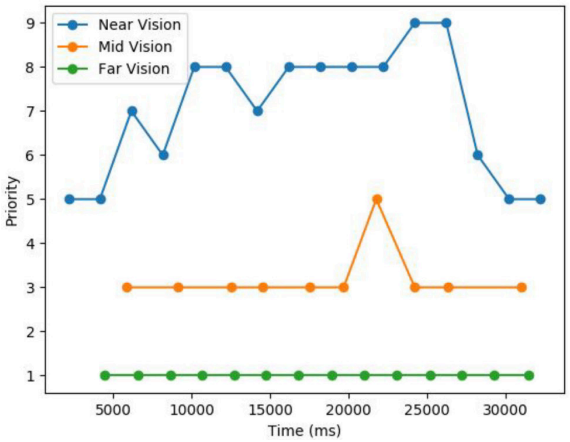

Figure 8: Experiment 3: sensors' priority - Mass 50.

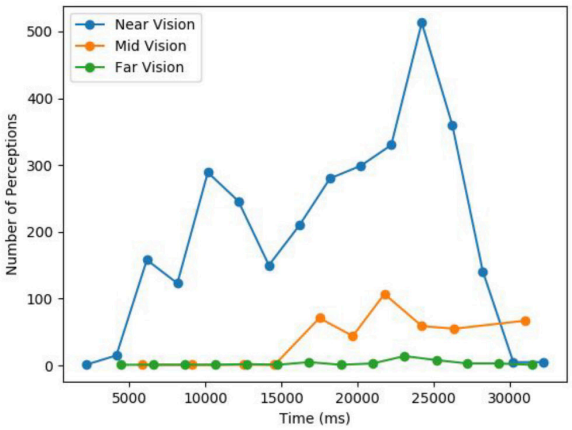

Figure 9: Experiment 3: sensors' number of perceptions - Mass 50.

Sensors behaviours are presented in figures 8 and 9. Based on these two figures, it is possible to notice the near vision sensor as the one with the most priority variation. It happens because the receiving number of perceptions was higher than the mass value. The other sensors were not receiving as many perceptions as the mass value, hence becoming more stable.

In the fourth experiment, the following research question is established: "What is the impact of having a mass with a high value when the sensor does not receive a large number of perceptions?". We choose the value of 80 to the mass, and the near vision sensor was normalized to evaluate the role of a higher mass. See Figure 10. Moreover, 11 shows that the sensors kept the priorities with the minimum value most of the time. Since the sensors were not receiving enough number of perceptions, they were stable. This behavior states that using great value as mass leads to low priority sensors.

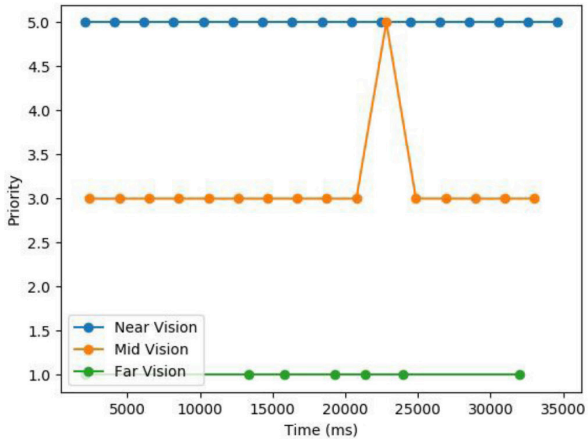

Figure 10: Experiment 4: sensors' priority - Mass 80.

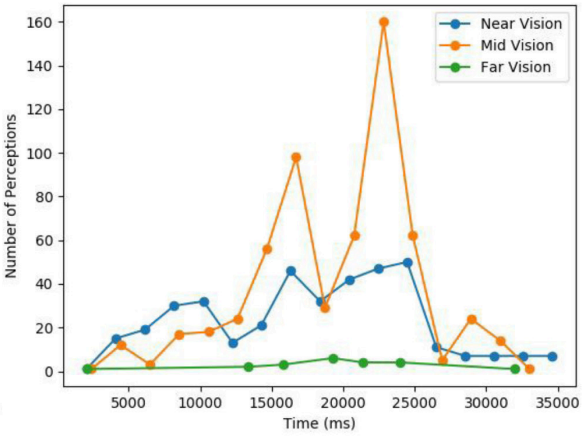

Figure 11: Experiment 4: sensors' number of perceptions - Mass 80.

5. Conclusion

Agents situated in dynamic environments may receive unnecessary information for its decision-making process. This information can be redundant or irrelevant to the agents' current goals. In this paper, we analyzed the process of perceptions in agents situated in a virtual environment.

Based on the characteristics of virtual environments, we developed perceptions policies for sensors. The sensors' policies allow the agent to process less information, working as filters. This approach allows the sensors to capture a higher number of perceptions, once the agent does not process all this information.

We present two policies for the sensors' prioritization: progressive and sudden. The first increases/decreases the sensors' priority based on the number of new information; Moreover, the next focus on the amount of perception arriving at a specific period. The progressive strategy is acceptable in virtual games, where the priority of their sensors is less critical. We state that health care environments, for example, can use the sudden strategy, in which sudden changes may trigger an alert.

Four experiments were conducted to evaluate the proposed policies. Two of them aimed to evaluate the progressive policy with and without considering the agents' reasoning cycle and the two others aimed to evaluate the role of mass and number of perceptions in the sudden policy.

5.1. Future work

We are working now to improve the process of sensing in intelligent agents. Our next step is to integrate the progressive and sudden policies with the active perception components defined by Bajcsy *et al.* (Bajcsy et al., 2018) as discussed in the section 2. The literature already has some analysis of the integration of active perception with BDI agents reasoning, but we plan to explore the use of Sigon agents and investigate the role of multiple knowledge representations.

From a practical application perspective, it is interesting to evaluate the use of the UDP protocol instead of TCP for agent communication with a virtual environment. However, as the UDP protocol does not have all the guarantees of the TCP, the communication between agent and environment would not be suspended, and the agent could lose some perceptions.

Acknowledgment: This study was financed in part by the Coordination for the Improvement of Higher Education Personnel - Brazil (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - CAPES) - Finance Code 001.

6. References

Bajcsy, R., Aloimonos, Y., and Tsotsos, J. K., 2018. Revisiting active perception. *Autonomous Robots*, 42(2):177–196.

Brewka, G., Eiter, T., and Fink, M., 2011. Nonmonotonic multi-context systems: A flexible approach for integrating heterogeneous knowledge sources. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6565 LNAI:233–258. ISSN 03029743. doi:10.1007/978-3-642-20832-4_16.

Casali, A., Godo, L., and Sierra, C., 2005. *Graded BDI Models for Agent Architectures*, volume 3487.

Corchado, J. M., Bajo, J., De Paz, Y., and Tapia, D. I., 2008. Intelligent environment for monitoring Alzheimer patients, agent technology for health care. *Decision Support Systems*, 44(2):382–396.

Endsley, M. R., 1988. Situation awareness global assessment technique (SAGAT). In *Aerospace and Electronics Conference, 1988. NAECON 1988., Proceedings of the IEEE 1988 National*, pages 789–795. IEEE.

Farias, G., Dimuro, G., and Rocha Costa, A., 2010. BDI agents with fuzzy perception for simulating decision making in environments with imperfect information. volume 627.

Gelaim, T. Â., Hofer, V. L., Marchi, J., and Silveira, R. A., 2019. Sigon: A multi-context system framework for intelligent agents. *Expert Systems with Applications*, 119:51–60.

Gibson, J. J., 1950. The perception of the visual world.

Giunchiglia, F. and Serafini, L., 1994. Multilanguage hierarchical logics, or: how we can do without modal logics. *Artificial intelligence*, 65(1):29–70.

Herrero, P., Greenhalgh, C., and De Antonio, A., 2005. Modelling the sensory abilities of intelligent virtual agents. *Autonomous Agents and Multi-Agent Systems*, 11(3):361–385. ISSN 13872532. doi:10.1007/s10458-005-2921-8.

Mello, R. R. P., Gelaim, T. A., and Silveira, R. A., 2018. Negotiating Agents: A Model Based on BDI Architecture and Multi-Context Systems Using Aspiration Adaptation Theory as a Negotiation Strategy. In *Proceedings of the 12th International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS-2018)*. Springer International Publishing, Matsue, Japan.

Rafaeli, N. and Kaminka, G. A., 2017. Active Perception at the Architecture Level. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, pages 1708–1710. International Foundation for Autonomous Agents and Multiagent Systems.

So, R. and Sonenberg, L., 2009. The Roles of Active Perception in Intelligent Agent Systems. In *Multi-Agent Systems for Society: 8th Pacific Rim International Workshop on Multi-Agents, PRIMA 2005, Kuala Lumpur, Malaysia, September 26-28, 2005, Revised Selected Papers*, volume 4078, page 139. Springer Science & Business Media.

Stabile, J., M.F. and Sichman, J., 2016. Evaluating Perception Filters in BDI Jason Agents. pages 116–121. doi:10.1109/BRACIS.2015.18. Unity Technologies, 2019. *Unity 3D*.

Van Oijen, J. and Dignum, F., 2011. Scalable perception for BDI-agents embodied in virtual environments. volume 2, pages 46–53. doi:10.1109/WI-IAT.2011.176.

Weyns, D., Steegmans, E., and Holvoet, T., 2004. Towards active perception in situated multi-agent systems. *Applied Artificial Intelligence*, 18(9-10):867–883.