# Virtual Agent Perception in Large Scale Multi-Agent Based Simulation Systems

# (Extended Abstract)

Dane Kuiper
University of Texas at Dallas
800 West Campbell Road
Richardson, Texas, USA
kuiper@utdallas.edu

Rym Z. Wenkstern
University of Texas at Dallas
800 West Campbell Road
Richardson, Texas, USA
rymw@utdallas.edu

## ABSTRACT

In this paper we discuss virtual agent perception in large scale open environment based MABS.

## Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Multiagent systems; I.6.8 [**Simulation and Modeling**]: Types of Simulation—*distributed*

## General Terms

Algorithms, Performance, Design, Experimentation

## Keywords

multi-agent simulation, virtual agent perception

## 1. INTRODUCTION

A perception system that models human sensory systems is critical for simulating virtual agents evolving in open environments (i.e., inaccessible, non-deterministic, dynamic, continuous). With respect to vision, until recently, very few realistic virtual agent vision perception techniques have been discussed in the literature. Most MABS have tackled the vision perception problem by providing agents with global environmental knowledge. Even though this approach is straightforward and easy to implement, it is unfit to simulate realistic scenarios. In this paper, we present efficient vision and vision obstruction algorithms, which are entirely implemented within the DIVAs Framework. We run experimentations regarding efficiency of the algorithms, and present and evaluate our results.

## 2. SURVEY

Perception, specifically the vision sense, plays an important role in realistic multi-agent simulations. Researchers and developers have approached the vision problem from three various perspectives. The most common approach consists of providing agents with global environmental knowledge [2]. Another approach consists of providing agents with environmental global knowledge at initialization time, and then allowing them to dynamically perceive information and act on their perceptions. Finally, the third approach consists of not allowing agents to access any global knowledge, and requiring them to make all decisions based on information perceived in simulated real-time. [3, 4]

There has also been extensive agent perception work on synthetic vision. However the nature of the problem is different. Synthetic vision algorithms focus on extracting data from images. Our work is unique in that our system provides the agents with no access to any global knowledge and requires all agents to perceive their environment through the perception module in order to gain knowledge. Our vision and vision obstruction algorithms are designed for use with large-scale open environments within the DIVAs framework and are fully configurable regarding desired accuracy.

**Cite as:** Virtual Agent Perception in Large Scale Multi-Agent Based Simulation Systems (Extended Abstract), Dane Kuiper and Rym Z. Wenkstern, *Proc. of 10th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2011)*, Tumer, Yolum, Sonenberg and Stone (eds.), May, 2–6, 2011, Taipei, Taiwan, pp. 1235-1236.

## 3. DIVAS VISION ALGORITHM

DIVAs (**D**ynamic **I**nformation **V**isualization of **A**gent **s**ystems) is a large scale distributed multi-agent system framework for the specification and execution of large scale distributed simulations where agents are situated in an open environment [1]. In this section we discuss a new algorithm and implementation of a vision sensor module for the DIVAs perception system. The main goal is to create a visual perception sensor module that is very efficient while continuing to provide realistic and useful data. The user has full control over how accurate the visual perception module can be. Accuracy is controlled via user settings for bounding boxes and our approximated vision cone. Bounding boxes can save many calculations by approximating highly complex objects. This is done by creating a simple box around the object and using that box for testing. Using our approximated vision cone also greatly increases efficiency.

### 3.1 The DIVAs Vision Algorithm

The DIVAs Vision Algorithm uses a set of steps to determine if an item is visible.

**Step (1)** Run the DIVAs Vision Algorithm to test all objects received from the environment to see if they are possibly visible.

**Step (2)** Run the DIVAs Obstruction Algorithm on objects that are possibly visible from step (1).

Due to space requirements, we will focus on the DIVAs Obstruction Algorithm.

## 4. DIVAS OBSTRUCTION ALGORITHM

An obstruction is an item that prevents another item from being seen. Based on the DIVAs Vision algorithm, each agent's vision is determined by its vision cone. But if, for example, there is a large wall directly in front of the agent, this wall would prevent the agent from seeing anything behind it.

With the DIVAs Vision algorithm, any object situated within the agent's cone of vision is visible by the agent, even if it is obstructed by a larger object. Hence, to implement a realistic vision system, it is necessary to complement our vision algorithm with an obstruction processing procedure. One simple way of addressing obstruction is to check if each item is being obstructed by every other item. Let $\lambda$ be the number of items. The algorithm is:

```
For ($i = 0$ to $\lambda$){For ($j = 0$ to $\lambda$){Check-Obstruct($i$,$j$)}}
```

We refer to this algorithm as *Basic Obstruction Algorithm.* This simple algorithm runs in time $c \cdot \lambda^2$, where c = the time to check an obstruction. Obviously in a large simulation with 1000s or more agents, a $\lambda^2$ algorithm is not ideal. The DIVAs Vision Obstruction Algorithm takes advantage of the following properties of obstructions in order to improve speed:

1. Only visible items need to be tested for obstruction. Something an agent cannot see is unable to obstruct the agent from seeing anything. Hence, we first run the basic DIVAs Vision algorithm to determine which items are even worth considering. This improves the algorithm runtime from $\lambda^2$ to $x^2$ where $x$ is the number of items returned by the basic DIVAs Vision algorithm.

2. Check the nearest and furthest points of each item. If an item's nearest point is further than another item's furthest point, then the further item cannot obstruct the nearer item.

3. Remove items that have already been shown to be obstructed from further testing. If it is known that item A is obstructed by larger item B, than checking if item A obstructs anything is not necessary.

However, even with these optimizations, a "perfect" vision algorithm would be extremely slow. So while we utilize these optimizations, the DIVAs Obstruction Algorithm also utilizes a cone-based line segment algorithm with Bounding Boxes around items. Each line segment in the vision cone will always intersect the closest item first. Any intersections beyond this first intersection are meaningless, since the first item intersected obstructs any further vision.

We now introduce the DIVAs Vision Obstruction Algorithm.

1. Run the DIVAs Vision algorithm on the $\lambda$ items. This algorithm returns a list of probable visible items $X[n]$, where $n$ is the number of items that will considered for obstruction testing. In our testing, $n$ is usually much smaller than $\lambda$.

2. For each of these $n$ items in $X$, calculate their Bounding Box.

3. Continue by calculating the line segments of the approximate agent's vision cone. We then check each line segment against each of the 6 Bounding Box faces of every item. For each line segment, only the closest item intersected is saved. At the end, this item is considered visible. Any item without any intersections is obstructed and not visible.

4. Test *one* extra line segment directly at the center of each item that is not hit by any line segments. During earlier calculations, the agent keeps a list of all items that have not been intersected by *any* line segment. After finishing all earlier steps, only then is the single extra line segment used. This step will eliminate most falsely obstructed small items. The time added by this step is usually constant (0-5 extra line segments are used.)

### 4.1 Obstruction Algorithm Results

In order to evaluate the strengths of the DIVAs Obstruction Algorithm, we run the Basic Obstruction Algorithm and the DIVAs Obstruction Algorithm on a DIVAs environment structured with a self-organizing cell hierarchy. We compared results using the number of intersection tests as a measure. We found through extensive testing that our approximation retained near perfect results while increasing speed by up to 9800%.

## 5. CONCLUSION

This paper has discussed Virtual Agent perception in large scale MABS. We covered efficient techniques for calculating vision and vision obstruction for Virtual Agents. Our results showed that efficiency improved by over 9800% in some scenarios. Future work includes continuing to increase efficiency of current vision techniques. Since our vision and obstruction algorithms are user configurable for accuracy, we would like to develop an automated accuracy balancing method. For example, depending on available processing resources, the system could either increase or decrease accuracy of the approximation.

## 6. ACKNOWLEDGMENTS

DIVAs supported by Rockwell-Collins grant 5-25143.

## 7. REFERENCES

[1] Multi-agent and visualization systems lab. http://mavs.utdallas.edu/. Accessed January 2011.

[2] B. Banerjee, A. Abukmail, and L. Kraemer. Advancing the layered approach to agent-based crowd simulation. In *Proceedings of IEEE Workshop on Parallel and Distributed Simulation*, pages 185–192, Rome, Italy, June 3-6 2008.

[3] N. Pelechano, J. Allbeck, and N. Badler. Controlling individual agents in high-density crowd simulation. In *2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 99–108, San Diego, California, August 02 - 04 2007.

[4] S. J. Rymill and N. A. Dodgson. Psychologically-based vision and attention for the simulation of human behaviour. In *Proceedings of Computer graphics and interactive techniques*, pages 229–236, Dunedin, New Zealand, November 29 - December 02 2005.