ABSTRACT
We consider multi-agent scenarios where each agent controls a surveillance camera positioned in the plane, with fixed position and angle of view, but rotating freely. The agents can thus observe the surroundings and each other. They can also reason about each other’s observation abilities and knowledge derived from these observations.

In this demonstration, cameras are located in the plane. The user can interact with the cameras, check epistemic properties and announce formulas. The camera can also turn in order to satisfy an epistemic property.

1. INTRODUCTION
Modeling and study of multi-agent systems that involve intelligent agents combining perceptual and reasoning abilities is a major field of research and applications of AI. In particular, in camera surveillance, we want to check some properties as:

1. camera $a_1$ knows that camera $a_3$ sees the intruder $b$ or camera $a_2$ knows that camera $a_3$ sees the intruder $b$ (this is called distributed knowledge about $a_1$ and $a_2$ that camera $a_3$ sees the intruder $b$);
2. camera $a_1$ knows that camera $a_2$ knows that camera $a_1$ knows etc. that camera $a_3$ sees the intruder $b$ (this is called common knowledge about $a_1$ and $a_2$ that camera $a_3$ sees the intruder $b$).

First we describe the demonstration. Then we recall the variant of epistemic modal logic used here. This is maybe the first real demonstration involving epistemic modal logic.

2. DEMONSTRATION

2.1 Overview
We propose to dispose cameras on a table (some of them have hats) and a red ball representing the intruder. The system can then check properties on the current situation (model checking) or automatically turn the cameras for satisfying a given formula (satisfiability problem). In particular, we are able to simulate the muddy children puzzle and the prisoners' puzzle. The reader can find information about the demonstration here:


2.2 Interactive aspects
During the demonstration, the user can interact with the system in five possible ways:

1. She can turn the cameras by hand;
2. She can move the red ball;
3. She can add/remove hats to cameras;
4. She can enter a property to check in epistemic modal logic described in the next section;
5. After the positions of the cameras are fixed, the position of the ball is fixed and the hats are fixed, she can make public announcements of a property $\phi$.

The interaction is divided in two phases:

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initialization</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Communication</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

2.3 Technical details
A camera is built up with:

- A position-controlled actuator (Robotis Dynamixel RX24F or equivalent);
- A webcam (C310 HD Webcam Usb 2.0, 1280x720 pixels).

A serial converter interface connects the actuators to the computer.

2.4 Architecture
In real life, cameras are autonomous: if a camera $a_1$ sees another camera $a_2$, $a_1$ should be able to infer the direction of view of $a_2$ from the image returned by the webcam of $a_1$. 
Here, in this demonstration, the system will not have a distributed architecture. For technical reasons, the knowledge of the cameras is computed externally by the computer that has access to the directions of view of all the cameras.

Nevertheless, the position of the ball and hats are inferred from images returned by the webcam. The following diagram shows the architecture around the model checking procedure:

On the contrary, the satisfiability problem procedure will modify the angles of the cameras in order to satisfy a specification. The following diagram shows the architecture around the procedure for solving the satisfiability problem:

3. EPISTEMIC MODAL LOGIC

We recall the framework that is presented in more details in [1].

3.1 Language

Formulas are built up from the following primitives: The set of well-formed formulas we can check is given by the following BNF:

\[
\begin{align*}
\varphi &::= a_1 \triangleright a_2 \quad \text{camera } a_1 \text{ sees camera } a_2 \\
\varphi &::= a > b \quad \text{camera } a \text{ sees the red ball } b \\
\varphi &::= \text{hat}_a \quad \text{camera } a \text{ wears a hat} \\
\neg \varphi &::= \varphi \text{ is false} \\
\varphi \lor \psi &::= \varphi \text{ or } \psi \\
K \varphi &::= \text{camera } a \text{ knows } \varphi \\
C \varphi &::= \text{cameras in } J \text{ commonly know that } \varphi
\end{align*}
\]

The semantics is given in terms of a Kripke model \( M_0 \) made up of all possible angle assignments to the cameras. For more information about the semantics, the reader may refer to [1].

3.2 Model checking

The positions of the cameras are fixed and we first compute the so-called vision sets, that is, for a given camera \( a \), the set of all possible sets of cameras that \( a \) can see.

The model checking is implemented as follows: from the vision sets and the set of cameras that see the red ball, we browse the inferred Kripke model on the fly and we evaluate the formula. For more information about the model checking procedure, the reader may refer to [1].

About public announcements.

At the beginning of the demonstration, the model is \( M_0, w \) where \( w \) is the actual world. Then, the user can announce a true formula \( \varphi \) in the current situation. The current model \( M \) is replaced by the updated model \( M^\varphi \) that is the subgraph of \( M \) made up of the worlds \( u \) such that \( M, u \models \varphi \) [2].

The algorithm is an adaptation of the algorithm of model checking for a variant of this framework [3].

3.3 Satisfiability problem

The satisfiability problem consists in turning the cameras so that a given property is satisfied. We here restrict the language by avoiding constructions \( a > b \) since we can not move the ball. For more information about the procedure for the satisfiability problem, the reader may refer to [1].

4. CONCLUSION

In this demonstration, we argue the feasibility of using epistemic modal logic to specify a system of cameras and to check properties.

Up to now, we have two phases during the interaction: the initialisation phase and the communication phase. The initialisation phase is made up of ontic actions whereas the communication phase is made up of pure communicative actions. In order to being able to mix ontic and communicative actions, we plan to allow use revision instead of public announcement and belief instead of knowledge. Another future work consist in handling mobile agents. Efficient algorithms for such features are not yet completely established. As a long-term project, we plan to build a logical framework for planning involving temporal and epistemic properties (that is, epistemic properties may be invariants, objectives etc.).

Acknowledgements.

We warmly thank Olivier Gasquet and Valentin Goranko for their comments. This work is supported by the Collège de Recherche Hubert Curien at ENS Rennes, France.

5. REFERENCES

