Big Brother Logic: Logical modeling and reasoning about agents equipped with surveillance cameras in the plane

Technical report IRIT/RT–2014-01–FR

Olivier Gasquet
University Toulouse III - Paul Sabatier
IRIT, France
Olivier.Gasquet@irit.fr

Valentin Goranko
Technical University of Denmark
Kgs. Lyngby, Denmark
vfgo@imm.dtu.dk

Francois Schwarzentruber
ENS Rennes
schwarze@ens-rennes.fr

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. We introduce suitable logical languages for reasoning about such scenarios which involve atomic formulae stating what agents can see, multi-agent epistemic operators for individual, distributed and common knowledge, as well as dynamic operators reflecting the ability of cameras to turn around in order to reach positions satisfying formulae in the language. We introduce 3 different but equivalent versions of the semantics for these languages, discuss their expressiveness and provide translations in PDL style. Using these translations we develop algorithms and obtain complexity results for model checking and satisfiability testing for the basic logic BBL that we introduce here and for some of its extensions. Notably, we show that even for the extension with common knowledge, model checking remains in PSPACE. Finally, we discuss some further extensions: by adding obstacles, positioning the cameras in 3D or enabling them to change positions. Our work has potential applications to automated reasoning, formal specification and verification of observational abilities and knowledge of multi-robot systems.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Intelligent agents, Multiagent systems*; I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods—*Modal logic*

General Terms

Theory, Algorithms, Verification

This work has been partially supported by ANR-11-LABX-0040-CIMI within the program ANR-11-IDEX-0002-02

Keywords

logical reasoning, multi-agent systems, surveillance cameras, observational abilities, knowledge, model checking, satisfiability testing

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. An important issue in that field is the analytic processing of visual perception and visual communication, where the main problem is how to represent and reason logically about the visual information that the intelligent agents receive from the environment and from each other. This creates not only agents' knowledge about the surrounding world, but also about each other's knowledge about that world, etc. and that knowledge affects and guides their intelligent behavior.

In this paper we address these issues by considering multi-agent scenarios where each agent controls a stationary surveillance camera positioned in the plane. In the basic framework presented here each camera has a fixed position and angle of view, but can rotate freely around its axis. Through the cameras, the agents can observe the surrounding world and each other. We adopt some basic assumptions about these scenarios, in order to simplify or idealize them:

▷ We identify, conceptually and physically, the agents with their cameras. Thus, we can talk about agents seeing objects and about cameras' knowledge.

▷ We assume that the cameras/agents are positioned at single points in the plane, and are transparent. These assumptions can be easily relaxed and non-transparent cameras/agents of real sizes can be considered, without major modifications of the framework and the obtained results, by simply treating them both as cameras and as obstacles.

▷ We assume that the only objects of interest for any agent in our scenarios are the other cameras/agents. This assumption is not practically restrictive, because one can assume that all other objects of interest are agents equipped with cameras, too, which cannot see anything or we are simply not interested in what they can see and know.

▷ Here we also assume that the agents and their cameras are stationary. In reality these are often mobile. Adding mobility of cameras leads to models where only the set of agents and the vision angles of their cameras are specified, but not their positions. We leave the model checking and satisfiability problems for the case of mobile cameras to a follow-up work.

$\triangleright$ In the stationary setup we assume that every agent knows the exact positions (but not the directions of vision) of all agents/cameras, even of those that the agent does not see currently. This assumption is certainly well justified if agents are endowed with some memory, as they can turn around to observe and memorize the positions of the others.

We introduce suitable logical languages for reasoning about such scenarios which involve atomic formulae stating what agents can see, multi-agent epistemic operators for individual, distributed and common knowledge, as well as dynamic operators reflecting the ability of cameras to turn around in order to reach positions satisfying formulae in the language.

Here we are interested not only in what agents can see, but also in the *knowledge* they can derive from their visual perception. The agents' knowledge that we model and study in our framework is about other agents' observation abilities and about their direct and iterated knowledge derived from such observations. A subtle issue arises here, of how vision and knowledge are related. At first glance, it seems that agents only know about other agents' knowledge what they can immediately derive from what they can see regarding the observational abilities of those other agents. We show that this is not quite so, as there is a non-trivial deductive aspect in the analytic knowledge that agents can acquire.

We introduce semantics for our logical languages on natural geometric models, as well as formal Kripke semantics for them in vision-based finite abstractions of the geometric models. We then discuss their expressiveness and provide translations for them in the style of Propositional Dynamic Logic PDL [9]. Using these translations we develop algorithms and obtain complexity results for model checking and satisfiability testing for the basic logic BBL that we introduce here and for some of its extensions. The key observations that enables our model checking and satisfiability testing procedures is that, while the set of geometric models over a configuration of cameras with fixed positions is infinite, the set of their vision-based abstractions, obtained by identifying models where each agent can see the same configuration of agents, is finite.

Finally, we discuss some important extensions of the present work: adding obstacles, positioning the cameras in 3D or enabling them to change positions. Our work has potential applications to automated reasoning, formal specification and verification of observational abilities and knowledge of multi-robot systems.

We are aware of very few related previous studies on the topic, but not involving logical reasoning, e.g., [7], [4]. They have mainly considered technical, architectural and knowledge representational aspects of agent-based systems of multi-camera surveillance and information processing. To our knowledge, the only previous study related to logical reasoning in such scenarios is the one represented by [11] and [1], which is the main precursor of the present work.

The paper is organized as follows: after brief preliminaries in Section 2, we introduce the logics BBL and some extensions in Section 3, discuss their expressiveness and provide translations to PDL in Section 4 and develop algorithms and obtain complexity results for model checking and satisfiability testing in Section 5. We then introduce and discuss briefly some further extensions in Section 7 and end with the concluding Section 8.

2. PRELIMINARIES

2.1 Multi-agent epistemic models and logics

For the necessary background on modal logic refer e.g., to [3]. Here we only give the basics on multi-agent epistemic models and logics. For the conceptual aspects or further technical background refer to e.g., [6] or [14].

A *multi-agent epistemic structure* for a set of agents Agt is a tuple $\mathcal{S} = \langle \mathsf{Agt}, \mathsf{St}, \{\sim_{\mathsf{a}} \mid \mathsf{a} \in \mathsf{Agt}\}\rangle$, where St is the set of states (possible worlds) and $\sim_{\mathsf{a}}$ is the epistemic indistinguishability relation on St of the agent a. In our setup there will be no abstract atomic propositions, only concretely interpreted ones, so epistemic structures and models will coincide.

We consider fully expressive multi-agent epistemic logics, extending classical (propositional) logic by adding epistemic operators for *individual knowledge* K_{a}, for each agent a, and for *group knowledge* K_{A}, *distributed knowledge* D_{A}, and *common knowledge* C_{A} for every non-empty group of agents A.

Thus, the formulae of the language of the multi-agent epistemic logic MAEL are defined as follows:

$\varphi := p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \mathsf{K}_{\mathsf{a}}\varphi \mid \mathsf{K}_{\mathsf{A}}\varphi \mid \mathrm{D}_{\mathsf{A}}\varphi \mid \mathrm{C}_{\mathsf{A}}\varphi,$

where p ranges over a fixed set of atomic propositions PROP, $\mathsf{a} \in \mathsf{Agt}$ and A is a non-empty subset of Agt.

The formal semantics of the epistemic operators at a state in a multi-agent epistemic model $\mathcal{M} = (\mathsf{Agt}, \mathsf{St}, \{\sim_{\mathsf{a}} \mid \mathsf{a} \in \mathsf{Agt}\}, V)$ is given by the following truth definitions:

(K_{a}) $\mathcal{M}, q \models \mathsf{K}_{\mathsf{a}}\varphi$ iff $\mathcal{M}, q' \models \varphi$ for all q' such that $q \sim_{\mathsf{a}} q'$.

(K_{A}) $\mathcal{M}, q \models \mathsf{K}_{\mathsf{A}}\varphi$ iff $\mathcal{M}, q' \models \varphi$ for all q' such that $q \sim^{E}_{\mathsf{A}} q'$, where $\sim^{E}_{\mathsf{A}} = \bigcup_{\mathsf{a} \in \mathsf{A}} \sim_{\mathsf{a}}$.

(C_{A}) $\mathcal{M}, q \models \mathsf{C}_{\mathsf{A}}\varphi$ iff $\mathcal{M}, q' \models \varphi$ for all q' s.t. $q \sim^{C}_{\mathsf{A}} q'$, where $\sim^{C}_{\mathsf{A}}$ is the transitive closure of $\sim^{E}_{\mathsf{A}}$.

(D_{A}) $\mathcal{M}, q \models \mathsf{D}_{\mathsf{A}}\varphi$ iff $\mathcal{M}, q' \models \varphi$ for all q' such that $q \sim^{D}_{\mathsf{A}} q'$, where $\sim^{D}_{\mathsf{A}} = \bigcap_{\mathsf{a} \in \mathsf{A}} \sim_{\mathsf{a}}$.

2.2 Flatland

Interaction between visual perception and knowledge has been studied in [1], in a logical framework based on the following language, where a, b are agents:

$$(\mathcal{L}_{\triangleright,\mathsf{K}}) \quad \phi ::= a \triangleright b \mid \neg\phi \mid \phi \vee \phi \mid \mathsf{K}_a \phi$$

The atomic proposition $a \triangleright b$ reads "agent a sees agent b".

The language $\mathcal{L}_{\triangleright,\mathsf{K}}$ has no abstract atomic propositions. The primary intended models are geometric models where each agent is located at a point in the space and sees a half-space. Different agents are located at different points and every agent a who can see another agent b can also see b's direction of vision. An abstract semantics is based on Kripke models where the possible worlds are geometric models. The indistinguishably relation $\sim_{\mathsf{a}}$ for agent a in such Kripke model is defined as follows: two worlds w and u are such that $w \sim_{\mathsf{a}} u$ iff the agents seen by agent a are the same and are located in the same way in both w and u.

Two variants of this framework have been studied in [1]: Lineland, where the space is 1-dimensional, and Flatland, where the space is 2-dimensional, i.e., a plane. Figure 1 shows two possible worlds in Flatland that are indistinguishable for agent a, because they look the same in the parts

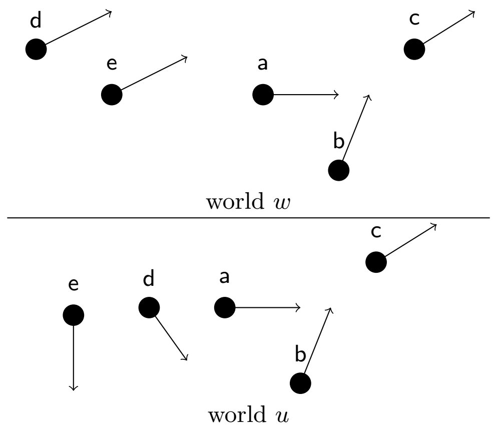


Figure 1: Two possible 2–dimensional worlds that are indistinguishable for agent a

which a can see. Note that a sees both b and c in both worlds, so a can see that b sees c, and this is the case in every world which is indistinguishable from these for a, hence a *knows* that b sees c. In fact, as shown in [11], $\mathsf{K}_a(b \rhd c)$ is equivalent to $a \rhd b \wedge a \rhd c \wedge b \rhd c$. On the other hand, a does not see d and e in any of these worlds, so a does not know whether d sees e in any of them. In fact, d sees e in world w and d does not see e in world u.

In that framework, the model checking problem takes as an input the description of a possible world w (positions and directions of agents) and a formula ϕ in $\mathcal{L}_{\rhd,\mathsf{K}}$ and asks whether ϕ holds in w. The satisfiability problem takes as an input a formula ϕ in $\mathcal{L}_{\rhd,\mathsf{K}}$ and asks whether there exists a world w satisfying ϕ. It has been proved in [1] that both model checking and satisfiability testing are PSPACE-complete for Lineland and are respectively PSPACE-hard and in EXPSPACE for Flatland.

2.3 First-order Theory of the Reals

Consider the following first-order language L_F for fields:

$$\phi ::= (e > 0) \mid (e = 0) \mid \neg\phi \mid (\phi \wedge \phi) \mid \forall x \phi \mid \exists x \phi$$

where e is a polynomial expression over variables $x, y, \ldots$. L_F has a standard interpretation in the field of reals $\mathbb{R}$. The problem of checking whether a closed formula of L_F is true in $\mathbb{R}$ is in EXPSPACE (see [13] for the historically first decision procedure and [2] for the description of an algorithm that runs in EXPSPACE) We will only need to solve that problem for the existential fragment of L_F, i.e., for sentences of the form $\exists x_1 \ldots \exists x_n \psi$ where ψ is open, for which the truth-checking problem in $\mathbb{R}$ is PSPACE-complete (see [5]) and we will use the algorithm described there in subsection 5.3.

3. LOGICS FOR AGENTS WITH STATIONARY SURVEILLANCE CAMERAS

Hereafter we fix a set of agents $\mathsf{Agt} = \{\mathsf{a}_1, \ldots, \mathsf{a}_k\}$.

Here we introduce our basic logic for stationary cameras, which we dub Big Brother Logic, or BBL for short.

3.1 Languages

The basic language of BBL involves the Flatland operator $\rhd$, the individual knowledge epistemic operators $\{\mathsf{K}_\mathsf{a}\}_{\mathsf{a} \in \mathsf{Agt}}$, plus the *turning* (diamond) operator $\langle \overset{\curvearrowright}{\mathsf{a}} \rangle$, where $\langle \overset{\curvearrowright}{\mathsf{a}} \rangle \phi$, means "a can turn around his position so that ϕ holds". Notation: $[\overset{\curvearrowright}{\mathsf{a}}] := \neg\langle \overset{\curvearrowright}{\mathsf{a}} \rangle \neg$. We also introduce the following extensions:

- $\mathsf{BBL_C}$: BBL plus all C_A for every group of agents A.
- $\mathsf{BBL_D}$: BBL plus all D_A for every group of agents A.
- Combinations of the above.

Additional operators: $\mathsf{a} \bowtie \mathsf{b}$, meaning "a sees b and b sees a" and $\mathsf{B}\,(\mathsf{abc})$, meaning "b is between a and c".

3.2 Semantics

We will introduce 3 different but equivalent types of models for the logic BBL.

3.2.1 Geometric models

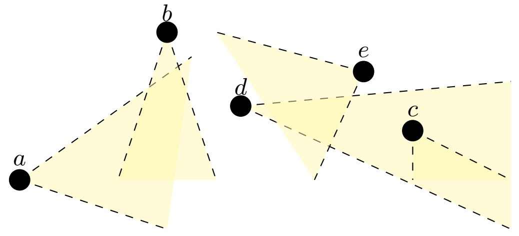
Camera configurations are formally represented by geometric models defined in the Euclidean plane $\mathbb{R}^2$ (or just a region) as follows. Every agent's camera is associated with a position (a point in the plane), an angle of view and a unit vector representing the direction of its vision. Formally:

Definition 3.1. *Let U be the set of unit vectors of $\mathbb{R}^2$. A geometric model is a tuple* $(\mathsf{pos}, \mathsf{dir}, \mathsf{ang})$ *where:*

- $\mathsf{pos} : \mathsf{Agt} \to \mathbb{R}^2$;
- $\mathsf{dir} : \mathsf{Agt} \to U$;
- $\mathsf{ang} : \mathsf{Agt} \to [0, 2\pi)$.

We denote $C_{p,u,\alpha}$ the sector that begins in point p from direction u turning in positive (counter-clockwise) direction, and has angle width α. We assume that different agents have different positions, every camera can "see" everything inside its sector of vision, including the positions and the directions of vision of all agents positioned there, but can see nothing outside of it. All cameras can turn at any angle.

Example 3.1 (Cameras and their sectors of vision).



3.2.2 Kripke structures and models

Now, we introduce Kripke structures in which the possible worlds are geometric models.

Definition 3.2. *Given* $\mathsf{pos} : \mathsf{Agt} \to \mathbb{R}^2$ *and* $\mathsf{ang} : \mathsf{Agt} \to [0, 2\pi)$*, we define a Kripke structure* $M = (\mathsf{pos}, \mathsf{ang}, D, R)$ *where D is the set of possible functions* $\mathsf{dir} : \mathsf{Agt} \to U$ *and R assigns to each agent* a *the following equivalence relation:*

$$R_\mathsf{a} = \{(\mathsf{dir}, \mathsf{dir}') \in D^2 \mid \textit{for all } \mathsf{b} \neq \mathsf{a},\ \mathsf{dir}(\mathsf{b}) = \mathsf{dir}'(\mathsf{b})\}$$

The truth of formulae is evaluated in such Kripke structures with additionally specified direction function $\mathsf{dir} \in D$. The truth conditions are defined as follows.

- $(\mathsf{pos}, \mathsf{ang}, D, R), \mathsf{dir} \models \mathsf{a} \rhd \mathsf{b}$ iff $\mathsf{pos}(\mathsf{b}) \in C_{\mathsf{pos(a)},\mathsf{dir(a)},\mathsf{ang(a)}}$;
- $(\mathsf{pos}, \mathsf{ang}, D, R), \mathsf{dir} \models \langle \overset{\curvearrowright}{\mathsf{a}} \rangle \phi$ iff there exists $\mathsf{dir}' \in R_\mathsf{a}(\mathsf{dir})$ such that $(\mathsf{pos}, \mathsf{ang}, D, R), \mathsf{dir}' \models \phi$.

We define knowledge as in the logic of Flatland: an agent a knows a property ϕ iff ϕ holds in all possible worlds that are compatible with what a sees. Those possible worlds can only differ from the actual world in the positions of the agents not seen by agent a. Formally, the semantics of all epistemic operators is given as in Section 2.1 by regarding the relation $\sim_{\mathsf{a}}$ as follows: $\mathsf{dir} \sim_{\mathsf{a}} \mathsf{dir}'$ iff the set of agents seen by a in dir and dir' are the same and they have the same directions of view. The difference with the approach described in subsection 2.2 is that here agents have common knowledge of the positions of all agents.

Note that if all agents seen by camera a are in the interior of its sector of vision, then a slight change of the direction of vision of a camera may not change what the respective agent sees, so all cameras would see the same objects if they are rotated at sufficiently small angles. This naturally defines an equivalence relation over worlds in pointed Kripke models where equivalent worlds satisfy the same $\triangleright$-propositions. Formally, given a Kripke structure $\mathcal{M} = (\mathsf{pos}, \mathsf{ang}, D, R)$ and $\mathsf{dir}, \mathsf{dir}' \in D$, we define:
$\mathsf{dir} \equiv \mathsf{dir}'$ iff $(\forall \mathsf{a}, \mathsf{b} \in \mathsf{Agt} : \mathcal{M}, \mathsf{dir} \models \mathsf{a} \triangleright \mathsf{b}$ iff $\mathcal{M}, \mathsf{dir}' \models \mathsf{a} \triangleright \mathsf{b})$. The equivalence class of dir with respect to $\equiv$ will be denoted by $\overline{\mathsf{dir}}$. Since Agt is finite, there are finitely many equivalence classes w.r.t. a given Kripke structure (see prop. 3.2).

Lemma 3.1. *Given any Kripke structure $\mathcal{M} = (\mathsf{pos}, \mathsf{ang}, D, R)$, for every agent a we have that $(\equiv \circ R_{\mathsf{a}}) = (R_{\mathsf{a}} \circ \equiv)$, where $\circ$ is the composition of relations.*

Proof. Let $(\mathsf{dir}, \mathsf{dir}') \in (\equiv \circ R_{\mathsf{a}})$, and let dir'' be the same as dir but with $\mathsf{dir}''(\mathsf{a}) = \mathsf{dir}'(\mathsf{a})$. Then $(\mathsf{dir}, \mathsf{dir}'') \in R_{\mathsf{a}}$ and $\mathsf{dir}'' \equiv \mathsf{dir}'$. Similarly the other way round. $\square$

All these Kripke structures are infinite. We give below an equivalent definition for the semantics based on their quotients w.r.t the equivalence relation $\equiv$.

3.2.3 Direction-based and vision-based abstractions of Kripke models

Given a Kripke structure $\mathcal{M} = (\mathsf{pos}, \mathsf{ang}, D, R)$ the *direction-based abstraction of* $\mathcal{M}$ is the Kripke model $\mathcal{K}_{\mathcal{M}} = (\overline{W}, \overline{R}, s)$ with valuation s as follows:

- $\overline{W} = \{\overline{\mathsf{dir}} \mid \mathsf{dir} \in D\}$;
- $\overline{R}$ maps each agent a to the equivalence relation $\overline{R}_{\mathsf{a}} = \{(\overline{\mathsf{dir}}, \overline{\mathsf{dir}'}) \mid (\mathsf{dir}, \mathsf{dir}') \in (\equiv \circ R_{\mathsf{a}})\}$;
- for all $\overline{w} \in \overline{W}$: $\overline{w} \in s(\mathsf{a} \triangleright \mathsf{b})$ iff $\mathcal{M}, w \models \mathsf{a} \triangleright \mathsf{b}$.

Proposition 3.1. *Let $G = (\mathsf{pos}, \mathsf{dir}, \mathsf{ang})$ be a geometric model, $\mathcal{M}$ be the Kripke structure extracted from G and $\mathcal{K}_{\mathcal{M}}$ be the direction-based abstraction of $\mathcal{M}$. Then the mapping $h : D \to W$ defined by $h(\mathsf{dir}) = \overline{\mathsf{dir}}$ is a bounded morphism from $\mathcal{M}$ onto $\mathcal{K}_{\mathcal{M}}$, and consequently, a bisimulation.*

Proof. We check the bisimulation conditions:

Atomic harmony: $\mathcal{M}, \mathsf{dir} \models \mathsf{a} \triangleright \mathsf{b}$ iff $\mathcal{K}_{\mathcal{M}}, \overline{\mathsf{dir}} \models \mathsf{a} \triangleright \mathsf{b}$ for any $\mathsf{dir} \in D$, by definition.

Forth: If $\mathsf{dir}, \mathsf{dir}' \in D$ and $(\mathsf{dir}, \mathsf{dir}') \in R_{\mathsf{a}}$ then $(h(\mathsf{dir}), h(\mathsf{dir}')) \in \overline{R}_{\mathsf{a}}$ by definition of $\overline{R}_{\mathsf{a}}$.

Back: If $(h(\mathsf{dir}), w) \in \overline{R}_{\mathsf{a}}$ for some $w \in \overline{W}$ then $w = \overline{\mathsf{dir}'}$ for some $\mathsf{dir}' \in D$ such that $(\mathsf{dir}, \mathsf{dir}') \in (\equiv \circ R_{\mathsf{a}})$. Then there is $\mathsf{dir}'' \in D$ s.t. $\mathsf{dir} \equiv \mathsf{dir}''$, hence $h(\mathsf{dir}) = h(\mathsf{dir}'')$ and $(\mathsf{dir}'', \mathsf{dir}') \in R_{\mathsf{a}}$. $\square$

Corollary 3.1. *For all formulas $\phi \in \mathsf{BBL}$, we have that $\mathcal{M}, \mathsf{dir} \models \phi$ iff $\mathcal{K}_{\mathcal{M}}, \overline{\mathsf{dir}} \models \phi$.*

In fact, each $\overline{\mathsf{dir}} \in \overline{W}$ is entirely characterized by the possible families of sets – one for each camera – of other agents that the camera can see in a glance if suitably turned. Because of this, we can identify a possible world with the tuple stating for each agent the set of other agents that it sees. Of course, we can precompute theses tuples, and also for each agent a we can pre-compute the family S_{a} of the sets of agents that can be seen at the same time by a, by using standard analytic geometry (we omit the routine details).

Definition 3.3. *Formally (where $\mathsf{dir} \in D$):*

- $\Gamma_{\mathsf{a}}^{\overline{\mathsf{dir}}} = \{\mathsf{b} \in \mathsf{Agt} \mid \mathsf{pos}(\mathsf{b}) \in C_{\mathsf{pos}(\mathsf{a}),\mathsf{dir}(\mathsf{a}),\mathsf{ang}(\mathsf{a})}\}$ *is the set* a *sees in direction given by* $\overline{\mathsf{dir}}$*;*
- $(\Gamma_{\mathsf{a}}^{\overline{\mathsf{dir}}})_{\mathsf{a} \in \mathsf{A}}$ *is the tuple of sets of agents each agent sees in directions given by* $\overline{\mathsf{dir}}$*;*
- $S_{\mathsf{a}} = \{\Gamma_{\mathsf{a}}^{\overline{\mathsf{dir}}} \mid u \in U\}$*; is the family of all sets of agents* a *can see when turning.*

Example 3.2. *In the geometrical model in Example 3.1 we have (trust us about the angles):*

- $S_{\mathsf{a}} = \{\emptyset, \{\mathsf{b}\}, \{\mathsf{b}, \mathsf{d}\}, \{\mathsf{b}, \mathsf{d}, \mathsf{e}\}, \{\mathsf{d}, \mathsf{e}, \mathsf{c}\}, \{\mathsf{e}, \mathsf{c}\}, \{\mathsf{c}\}\})$
- $S_{\mathsf{b}} = \{\emptyset, \{\mathsf{a}\}, \{\mathsf{e}\}, \{\mathsf{e}, \mathsf{c}\}, \{\mathsf{c}, \mathsf{d}\}, \{\mathsf{d}\}\}$;
- $S_{\mathsf{c}} = \{\emptyset, \{\mathsf{a}\}, \{\mathsf{a}, \mathsf{d}\}, \{\mathsf{a}, \mathsf{d}, \mathsf{b}\}, \{\mathsf{d}, \mathsf{b}\}, \{\mathsf{b}, \mathsf{e}\}, \{\mathsf{e}\}\}$;
- $S_{\mathsf{d}} = \{\emptyset, \{\mathsf{a}\}, \{\mathsf{b}\}, \{\mathsf{e}\}, \{\mathsf{e}, \mathsf{c}\}, \{\mathsf{c}\}\}$;
- $S_{\mathsf{e}} = \{\emptyset, \{\mathsf{b}\}, \{\mathsf{b}, \mathsf{d}\}, \{\mathsf{b}, \mathsf{d}, \mathsf{a}\}, \{\mathsf{d}, \mathsf{a}\}, \{\mathsf{a}\}\}$.

Proposition 3.2. *Each set S_{a} contains at most $2k - 2$ sets of agents, where $k = (\#\mathsf{Agt})$ is the number of agents. Hence so does the set $\overline{R}_{\mathsf{a}}(\overline{\mathsf{dir}})$ in the vision-based abstract model.*

Proof. Let a be an agent and start turning the camera clockwise. The set of agents seen by a's camera only changes when one of the two boundary rays of the sector of vision of the camera passes through an agent. In one full circle each of these rays passes through $k - 1$ agents. $\square$

A direction-based abstract model can be equivalently defined as follows.

Definition 3.4. *A vision-based abstract model is a triple $N = (V, T, \sigma)$ where:*

- $V = \{(\Gamma_{\mathsf{a}}^{\overline{\mathsf{dir}}})_{\mathsf{a} \in \mathsf{Agt}} \mid$ *for all* $\mathsf{a} \in \mathsf{Agt}$, $\mathsf{dir} \in D\}$*;*
- T *maps each agent* a *to the equivalence relation* $T_{\mathsf{a}} = \{((\Gamma_{\mathsf{a}}^{\overline{\mathsf{dir}}})_{\mathsf{a} \in \mathsf{Agt}}, (\Gamma_{\mathsf{a}}^{\overline{\mathsf{dir}'}})_{\mathsf{a} \in \mathsf{Agt}}) \mid$ *for all* $\mathsf{b} \neq \mathsf{a}$, $\Gamma_{\mathsf{b}}^{\overline{\mathsf{dir}}} = \Gamma_{\mathsf{b}}^{\overline{\mathsf{dir}'}}\}$
- *for all* $(\Gamma_{\mathsf{a}}^{\overline{\mathsf{dir}}})_{\mathsf{a} \in \mathsf{Agt}} \in V$: $(\Gamma_{\mathsf{a}}^{\overline{\mathsf{dir}}})_{\mathsf{a} \in \mathsf{Agt}} \in \sigma(\mathsf{b} \triangleright \mathsf{c})$ *iff* $c \in \Gamma_{\mathsf{b}}^{\overline{\mathsf{dir}}}$.

The following can be shown by direct verification:

Proposition 3.3. *The mapping $h : \mathcal{K}_{\mathcal{M}} \to N$ defined by $h(\overline{\mathsf{dir}}) = (\Gamma_{\mathsf{a}}^{\overline{\mathsf{dir}}})_{\mathsf{a} \in \mathsf{A}}$ is an isomorphism between the direction-based abstract model $\mathcal{K}_{\mathcal{M}}$ and the vision-based model N.*

Corollary 3.2. *For all formulas $\phi \in \mathsf{BBL}$, we have that $\mathcal{K}_{\mathcal{M}}, \overline{\mathsf{dir}} \models \phi$ iff $N, (\Gamma_{\mathsf{a}}^{\overline{\mathsf{dir}}})_{\mathsf{a} \in \mathsf{A}} \models \phi$.*

We will henceforth denote possible worlds by dropping the superscript $\overline{\mathsf{dir}}$.

A BBL formula is *satisfiable* iff it is true in some geometric model (hence, in some abstract Kripke structure or in a vision-based abstraction); it is *valid* iff it is true in every geometric model.

4. EXPRESSIVENESS AND EXTENSIONS

4.1 Expressing properties and specifications

The language of BBL and its extensions can express various natural specifications regarding the visual abilities and knowledge of the agents. Here are some examples in BBL and in $\mathsf{BBL_{CD}}$:

• $\mathsf{a}\triangleright\mathsf{b} \rightarrow \mathsf{K}_a[\overset{\curvearrowright}{\mathsf{b}}](\mathsf{a}\triangleright\mathsf{b})$: "If a can see b, then a knows that whichever way b turns around, this will remain true."

• $(\mathsf{b}\triangleright\mathsf{a} \wedge \mathsf{b}\not\triangleright\mathsf{c}) \wedge \mathsf{K}_b\langle\overset{\curvearrowright}{\mathsf{a}}\rangle((\mathsf{a}\triangleright\mathsf{b} \wedge \mathsf{a}\triangleright\mathsf{c}) \rightarrow \hat{\mathsf{K}}_b\mathsf{K}_a(\mathsf{b}\triangleright\mathsf{a} \wedge \mathsf{b}\not\triangleright\mathsf{c})$: "If agent b sees a but not c and knows that a can turn around so as to see both b and c then b consider it possible that a knows that b sees a but not c."

• $(\mathsf{a} \bowtie \mathsf{b}) \rightarrow \mathsf{C}_{\mathsf{a,b}}(\mathsf{a} \bowtie \mathsf{b})$: "If a and b see each other, then this is a common knowledge amongst them."

• $\mathsf{K}_{\mathsf{a,b}}(\langle\overset{\curvearrowright}{\mathsf{a}}\rangle(\mathsf{a}\triangleright\mathsf{b}) \wedge \langle\overset{\curvearrowright}{\mathsf{b}}\rangle(\mathsf{b}\triangleright\mathsf{a})) \rightarrow \mathsf{C}_{\mathsf{a,b}}(\langle\overset{\curvearrowright}{\mathsf{a}}\rangle\langle\overset{\curvearrowright}{\mathsf{b}}\rangle(\mathsf{a} \bowtie \mathsf{b}))$. "If a and b know that each of them can turn around to see the other, then it is a common knowledge amongst them that they can both turn so as to see each other."

• $\langle\overset{\curvearrowright}{\mathsf{a}}\rangle(\mathsf{a}\triangleright\mathsf{c} \wedge \mathsf{a}\triangleright\mathsf{d}) \wedge \langle\overset{\curvearrowright}{\mathsf{b}}\rangle(\mathsf{b}\triangleright\mathsf{d} \wedge \mathsf{b}\triangleright\mathsf{e}) \rightarrow \langle\overset{\curvearrowright}{\mathsf{a}}\rangle\langle\overset{\curvearrowright}{\mathsf{b}}\rangle(\mathsf{D}_{\mathsf{a,b}}\mathsf{c}\triangleright\mathsf{e} \vee \mathsf{D}_{\mathsf{a,b}}\mathsf{c}\not\triangleright\mathsf{e})$. "If a can turn around so as to see both c and d and b can turn around so as to see both d and e then both a and b can turn around so as to make it a distributed knowledge amongst them whether c sees e." (The reader is invited to check by hand the validity of this formula.)

4.2 Defining betweenness and collinearity

Assuming that the cameras' angles of vision are strictly less that 2π, the relations of betweenness and collinearity can be expressed in the language of BBL as follows:

- The formula $[\overset{\curvearrowright}{\mathsf{a}}](\mathsf{a}\triangleright\mathsf{b} \rightarrow \mathsf{a}\triangleright\mathsf{c})$ is true in a geometric model precisely when c lies on the ray starting at a and passing through b, i.e., when b is between a and c or c is between a and b.

- Therefore, the formula $[\overset{\curvearrowright}{\mathsf{a}}](\mathsf{a}\triangleright\mathsf{b} \rightarrow \mathsf{a}\triangleright\mathsf{c}) \wedge [\overset{\curvearrowright}{\mathsf{b}}](\mathsf{b}\triangleright\mathsf{a} \rightarrow \mathsf{b}\triangleright\mathsf{c})$ is true in a geometric model precisely when c lies strictly between a and b. Thus, we can define

$$\mathsf{B}\,(\mathsf{acb}) := [\overset{\curvearrowright}{\mathsf{a}}](\mathsf{a}\triangleright\mathsf{b} \rightarrow \mathsf{a}\triangleright\mathsf{c}) \wedge [\overset{\curvearrowright}{\mathsf{b}}](\mathsf{b}\triangleright\mathsf{a} \rightarrow \mathsf{b}\triangleright\mathsf{c})$$

4.3 Relations between vision and knowledge

Clearly, seeing and visual knowledge are closely related: an agent knows that he sees whatever he sees: $\mathsf{a}\triangleright\mathsf{b} \rightarrow \mathsf{K_a a}\triangleright\mathsf{b}$ and knows that he does not see whatever he does not see: $\mathsf{a} \not\triangleright \mathsf{b} \rightarrow \mathsf{K_a a} \not\triangleright \mathsf{b}$. On the other hand, there is more in the agent's visual knowledge than what he can see directly. For instance, the following is valid for any agents $\mathsf{a}, \mathsf{b}, \mathsf{c}_1, \mathsf{c}_2, \mathsf{e}$, even if a does not see b:

$$\mathsf{a}\triangleright\mathsf{c}_1 \wedge \mathsf{a}\triangleright\mathsf{c}_2 \wedge \mathsf{B}\,(\mathsf{c}_1\mathsf{e}\mathsf{c}_2) \rightarrow \mathsf{K_a}(\mathsf{b}\triangleright\mathsf{c}_1 \wedge \mathsf{b}\triangleright\mathsf{c}_2 \rightarrow \mathsf{b}\triangleright\mathsf{e})$$

Thus, a has non-trivial (i.e., not logically valid) knowledge about b's visual abilities.

4.4 PDL-like form of BBL

Here we will adopt a different, PDL-like perspective on the logic BBL. For background on the propositional dynamic logic PDL refer e.g., to [9]. We can consider every sequence of (non-deterministic) turns of cameras as a program, where turning the camera of an agent a at an unspecified angle is an atomic program, denoted by $\overset{\curvearrowright}{\mathsf{a}}$. Besides, we can consider tests as atomic programs and build complex programs by applying compositions, unions and iterations to these, like in PDL. The technical advantage is that, as we will show, all knowledge operators can be represented in terms of suitable programs, and eventually the whole language of BBL and all of its extensions considered here can be translated to versions of PDL.

More precisely, let us first introduce a two-sorted language, denoted BBL^{PDL}, with sorts for formulae and for programs, as follows, where $\mathsf{a}, \mathsf{b} \in \mathsf{Agt}$.

$$\phi ::= \mathsf{a}\triangleright\mathsf{b} \mid \neg\phi \mid \phi \vee \phi \mid [\gamma]\phi; \quad \gamma ::= \overset{\curvearrowright}{\mathsf{a}} \mid \phi? \mid \gamma;\gamma \mid \gamma \cup \gamma$$

The intuitive meaning of the program operations is as in PDL, e.g., $\phi?$ is the program that tests whether ϕ is true or not, $\gamma_1;\gamma_2$ is the sequential composition of γ_1 and γ_2. $\gamma_1 \cup \gamma_2$ represents the non-deterministic choice between γ_1 and γ_2. The formula $[\gamma]\phi$ is read as "after all executions of γ, the property ϕ holds". We also define the dual operator $\langle\gamma\rangle$ by $\langle\gamma\rangle\phi := \neg[\gamma]\neg\phi$, meaning "there exists an execution of γ such that ϕ holds afterwards". The formal semantics in a vision-based abstraction model $M = (W, R)$ and a world $w \in W$ is defined as follows:

- $M, w \models [\overset{\curvearrowright}{\mathsf{a}}]\phi$ iff for all $u \in R_{\mathsf{a}}(w)$, $M, u \models \phi$;
- $M, w \models [\phi?]\psi$ iff $M, w \models \phi \rightarrow \psi$;
- $M, w \models [\gamma_1;\gamma_2]\phi$ iff $M, w \models [\gamma_1][\gamma_2]\phi$;
- $M, w \models [\gamma_1 \cup \gamma_2]\phi$ iff $M, w \models [\gamma_1]\phi$ or $M, w \models [\gamma_2]\phi$.

For instance, the program "if a does not see b then a turns" is represented by $\gamma = (\neg\mathsf{a}\triangleright\mathsf{b})?;\overset{\curvearrowright}{\mathsf{a}}$.

4.5 Expressing the knowledge operators

Now, we will show how to simulate knowledge operators in terms of programs in BBL^{PDL}.

4.5.1 Expressing the individual knowledge operators

The (individual) knowledge operator K_a can be simulated with the following program, where $\{\mathsf{b}_1, \ldots, \mathsf{b}_n\} = \mathsf{Agt} \setminus \{\mathsf{a}\}$:

$$\gamma_{\mathsf{a}}^K = \left(\mathsf{a}\triangleright\mathsf{b}_1? \cup (\mathsf{a} \not\triangleright\mathsf{b}_1?;\overset{\curvearrowright}{\mathsf{b}_1})\right);\ldots;\left(\mathsf{a}\triangleright\mathsf{b}_n? \cup (\mathsf{a} \not\triangleright\mathsf{b}_n?;\overset{\curvearrowright}{\mathsf{b}_n})\right)$$

Indeed, each program $\left(\mathsf{a}\triangleright\mathsf{b}_i? \cup (\mathsf{a} \not\triangleright\mathsf{b}_i?;\overset{\curvearrowright}{\mathsf{b}_i})\right)$ can change the direction of view of b_i if and only if a does not see b_i, except when b_i is a. Thus, the program γ_{a}^K may turn arbitrarily all agents, other than a, that are not seen by a, which reflects precisely the semantics of K_a. Group knowledge operator $\mathsf{K_A}$ is then simulated by the program: $\gamma_{\mathsf{A}}^K = \bigcup_{\mathsf{a}\in\mathsf{A}} \gamma_{\mathsf{a}}^K$.

4.5.2 Expressing distributed knowledge

The distributed knowledge operator $\mathsf{D_A}$ is simulated with the following program, where $\{\mathsf{b}_1, \ldots, \mathsf{b}_n\} = \mathsf{Agt} \setminus \mathsf{A}$:

$$\begin{aligned}\gamma_{\mathsf{A}}^D = \;& \left(\textstyle\bigvee_{\mathsf{a}\in\mathsf{A}} \mathsf{a}\triangleright\mathsf{b}_1? \bigcup \left(\textstyle\bigwedge_{\mathsf{a}\in\mathsf{A}} \mathsf{a} \not\triangleright\mathsf{b}_1?\right);\overset{\curvearrowright}{\mathsf{b}_1}\right);\ldots;\\ & \left(\textstyle\bigvee_{\mathsf{a}\in\mathsf{A}} \mathsf{a}\triangleright\mathsf{b}_n? \bigcup \left(\textstyle\bigwedge_{\mathsf{a}\in\mathsf{A}} \mathsf{a} \not\triangleright\mathsf{b}_n?\right);\overset{\curvearrowright}{\mathsf{b}_n}\right)\end{aligned}$$

The construction $\left(\bigvee_{\mathsf{a}\in\mathsf{A}} \mathsf{a}\triangleright\mathsf{b}_i? \bigcup \left(\bigwedge_{\mathsf{a}\in\mathsf{A}} \mathsf{a} \not\triangleright\mathsf{b}_i?\right);\overset{\curvearrowright}{\mathsf{b}_i}\right)$, as in the translation of individual knowledge operator, is a program that turns agent b_i if and only if no agent in A sees it. This reflects precisely the semantics of $\mathsf{D_A}$.

4.5.3 *Expressing common knowledge*

In order to simulate the common knowledge operator C_A, we need to add the iteration operator $*$ in the language, that is, if γ is a program then γ^* is the program that repeats γ any finite number of times. The resulting language is called BBL_{PDL^*}. The semantics is defined on a vision-based abstraction model $M = (W, R)$ and $w \in W$ as follows:

- $M, w \models [\gamma^*]\phi$ iff $M, w \models [\gamma]^n\psi$ for all integers $n \geq 0$, where $[\gamma]^0\phi = \phi$ and $[\gamma]^n\phi = [\gamma][\gamma]^{n-1}\phi$ for $n \geq 1$.

Common knowledge among a group $\mathsf{A} = \{\mathsf{a}_1, \dots, \mathsf{a}_k\}$ can be simulated by: $\gamma^C_\mathsf{A} = \left(\gamma_{\mathsf{a}_1}; \dots; \gamma_{\mathsf{a}_k}\right)^*$. As the program γ_{a_i} represents the epistemic relation for agent a_i, the program γ^C_A corresponds to taking an arbitrary finite path along the epistemic relations for all agents in A. Note that this simulation is correct because the programs γ_{a_i} are reflexive and commute with each other.

4.6 Translation into BBL^{PDL} and BBL_{PDL^*}

Let τ be the translation from BBL into BBL^{PDL} and BBL_{PDL^*} defined by: $\tau(\mathsf{a} \rhd \mathsf{b}) = \mathsf{a} \rhd \mathsf{b}$, $\tau(\phi_1 \vee \phi_2) = \tau(\phi_1) \vee \tau(\phi_2)$, $\tau(\neg\phi) = \neg\tau(\phi)$, $\tau(X_\alpha\phi) = \gamma^X_\alpha\tau(\phi)$ (X is K, D or C).

The next claim follows immediately from the constructions of the knowledge simulating programs.

PROPOSITION 4.1. *Let $N = (V, T)$ be a vision-based abstraction model (def. 3.4), and $v \in V$, then: $N, v \models \phi$ iff $N, v \models \tau(\phi)$.*

5. DECISION PROCEDURES

In this section, we address the model checking problem and the satisfiability problem for BBL and its extensions. For any of these languages $\mathcal{L}$, the model checking problem $\mathrm{MC}(\mathcal{L})$, is defined as follows:

Input: a geometric model $\mathsf{G} = (\mathsf{pos}, \mathsf{dir}, \mathsf{ang})$ and a formula ϕ of $\mathcal{L}$.

Output: 'yes' iff $\mathcal{K}_\mathcal{M}, w_\mathsf{G} \models \phi$ where $\mathcal{K}_\mathcal{M}$ is the vision-based abstraction of G.

The satisfiability problem $\mathrm{SAT}(\mathcal{L})$ is defined as follows.

Input: a formula ϕ in $\mathcal{L}$;

Output: 'yes' iff there exists a geometric model $\mathsf{G} = (\mathsf{pos}, \mathsf{dir}, \mathsf{ang})$ such that $\mathcal{K}_\mathcal{M}, w_\mathsf{G} \models \phi$ where $\mathcal{K}_\mathcal{M}$ is the vision-based abstraction of G.

5.1 Upper-bound for MC(BBL_{PDL^*})

The procedure for the model checking problems above consists in two stages: first compute sets S_a (def. 3.3) in the vision based abstraction model (see Definition 3.4) from the geometric model (see Definition 3.1) and then do model checking on worlds of the vision based abstraction model.

5.1.1 *Computing the vision-based abstraction model*

Let us consider a vision based abstraction model and an agent a. For computing the set S_a, we consider a polar coordinate system centered in $\mathsf{pos}(\mathsf{a})$ and we consider all agents $b \neq a$ as a list sorted by angles. We perform a scan of all agents $b \neq a$ and we compute the possible sets of agents whose positions are in a sector centered in $\mathsf{pos}(\mathsf{a})$ and of angle $\mathsf{ang}(\mathsf{a})$. The set of those sets is exactly S_a. This can be achieved in linear time in the size of the vision based abstraction model and the reader can find the pseudo-code in Section 6 . The idea is to identify the positions of the sector of vision of a where one of the boundary rays passes through some of the other agents, and determine the sets of agents seen by a in those positions.

5.1.2 *Model checking in the vision-based model*

Now, we design a procedure *mc* for model checking. The specification of $mc(\Gamma, \phi)$ is to return true iff $M, \Gamma \models \phi$ where $M = (W, R)$ is the vision-based abstraction model and $\Gamma \in W$. The routine *mc* is very standard and the only interesting case concerns the model checking of a formula of the form $[\gamma]\psi$ where γ is a program, given by a regular expression. This case relies on a subroutine *path?* whose specification is: $path?(\Gamma, \Gamma', \gamma)$ returns true iff there is a γ-path from Γ to Γ', i.e. a path in the graph (W, R) whose labels match the regular expression γ. For instance, if $\Gamma(R_\mathsf{a} \circ R_\mathsf{a} \circ R_\mathsf{b})\Gamma'$, then there is a $(\overset{\curvearrowright}{\mathsf{a}};(\overset{\curvearrowright}{\mathsf{a}};\overset{\curvearrowright}{\mathsf{b}})^*)$-path from Γ to Γ' and therefore $path?\big(\Gamma, \Gamma', (\overset{\curvearrowright}{\mathsf{a}};(\overset{\curvearrowright}{\mathsf{a}};\overset{\curvearrowright}{\mathsf{b}})^*)\big)$ is true.

We want to prove that this model checking procedure works in PSPACE. All cases are straightforward except that of $\gamma = \gamma_1^*$. In order to treat this case, we introduce a new construction of program: $\gamma = \gamma_1^\mathbf{m}$ where $\mathbf{m}$ is an integer written in binary. The procedure *path?* uses the following simple observation: If there is a γ_1^*-path from Γ to Γ' then there is a $\gamma_1^\mathbf{m}$-path, where $0 \leq \mathbf{m} \leq 2^{k^2}$ (recall that $k = \#\mathsf{Agt}$). Thus, for the case of γ_1^*, we look for the existence of a $\gamma_1^\mathbf{m}$-path from Γ to Γ' for some $\mathbf{m} \in \{0, \dots, 2^{k^2}\}$.

Now, for the case $\gamma_1^\mathbf{m}$ when $\mathbf{m} > 1$ we use the *Divide and Conquer* method and browse all possible 'intermediate' worlds Γ'' in a $\gamma_1^{\lfloor\frac{\mathbf{m}}{2}\rfloor}$-path from Γ to Γ', that is, there is a $\gamma_1^{\lceil\frac{\mathbf{m}}{2}\rceil}$-path from Γ to Γ'' and a $\gamma_1^\mathbf{m}$-path from Γ'' to Γ'.

The resulting model checking procedure runs in polynomial space. The reader can find the pseudo-code of *path?* and *mc* in Section 6 .

PROPOSITION 5.1. *MC(BBL_{PDL^*}) is in PSPACE.*

5.2 Lower-bound results for model checking

Unsurprisingly, the lower-bound is reached even *without* the star operator in the language:

PROPOSITION 5.2. *MC(BBL^{PDL}) is PSPACE-hard.*

PROOF. We will give a polynomial reduction of the satisfiability problem SAT-QBF for quantified Boolean formulas (QBF), which is PSPACE-complete [12], to $\mathrm{MC}(\mathsf{BBL}^{PDL})$. Consider a QBF $\exists p_1 \forall p_2 \dots Q_n p_n \psi(p_1, \dots, p_n)$ where Q_k is $\exists$ if k is odd and $\forall$ if k is even and where $\psi(p_1, \dots, p_n)$ is Boolean formula over the atomic propositions $p_1, \dots, p_n$. Let $\mathsf{Agt} = \{\mathsf{a}_0, \mathsf{a}_1, \dots, \mathsf{a}_m\}$. We build an instance (G, ϕ), where $\mathsf{G} = (\mathsf{pos}, \mathsf{dir}, \mathsf{ang})$, for $\mathrm{MC}(\mathsf{BBL}^{PDL})$ as follows:

- $\mathsf{pos}(\mathsf{a}_i) = (i, 0)$; $\mathsf{dir}(\mathsf{a}_i) = (-1, 0)$; $\mathsf{ang}(\mathsf{a}_i) = \frac{\pi}{4}$;
- $\phi = \langle\overset{\curvearrowright}{\mathsf{a}_1}\rangle[\overset{\curvearrowright}{\mathsf{a}_2}]\dots O_i\psi'$ where O_i is $\langle\overset{\curvearrowright}{a_i}\rangle$ if i is odd and $[\overset{\curvearrowright}{a_i}]$ if i is even and ψ' is the formula ψ in which we have replaced all occurrences of p_i by $\mathsf{a}_i \rhd \mathsf{a}_0$.

Since each of the cameras positioned as above can independently be turned so as to see a_0 and it can also be turned so as not to see a_0, the formula $\exists p_1 \forall p_2 \ldots Q_n p_n \psi(p_1, \ldots, p_n)$ is true in the quantified Boolean logic QBL iff $\mathcal{K}_{\mathcal{M}}, w_{\mathsf{G}} \models \phi$ where $\mathcal{K}_{\mathcal{M}}$ is the vision-based abstraction model of G. $\square$

THEOREM 5.1. *Let* BBL^- *be the fragment of* BBL *without turning operators but with the knowledge operators. Then* $MC(\mathsf{BBL}^-)$ *is PSPACE-hard.*

PROOF. The proof is similar to the proof of Proposition 5.2 (and to the proof for the MC problem in Lineland, in [1]). We provide a polynomial reduction of the SAT-QBF problem to MC(BBL^-) as follows. Consider a QBF formula $\exists p_1 \forall p_2 \ldots Q_n p_n \psi(p_1, \ldots, p_n)$ where $Q_k = \exists$ if k is odd and $Q_k = \forall$ if k is even and where $\psi(p_1, \ldots, p_n)$ is Boolean formula over the atomic propositions $p_1, \ldots, p_n$. We build an instance $\mathsf{G} = (\mathsf{pos}, \mathsf{dir}, \mathsf{ang}), \phi$ for MC(BBL^-) as follows:

- $\mathsf{pos}(\mathsf{a}_i) = (2i, 0)$; $\mathsf{pos}(\mathsf{b}_i) = (2i+1, 0)$;
- $\mathsf{dir}(\mathsf{a}_i) = (-1, 0)$; $\mathsf{dir}(\mathsf{b}_i) = (-1, 0)$;
- $\mathsf{ang}(\mathsf{a}_i) = \frac{\pi}{4}$; $\mathsf{ang}(\mathsf{b}_i) = \frac{\pi}{4}$;
- $\phi = \hat{\mathsf{K}}_{\mathsf{a}_1}(\mathsf{a}_2 \triangleright \mathsf{a}_1 \wedge \mathsf{K}_{\mathsf{a}_2}(\mathsf{a}_3 \triangleright \mathsf{a}_1 \rightarrow \ldots \ldots \psi') \ldots)$ where ψ' is the formula ψ in which we have replaced all occurrences of p_i by $\mathsf{b}_i \triangleright \mathsf{a}_1$.

Again, we claim that the formula $\exists p_1 \forall p_2 \ldots Q_n p_n \psi(p_1, \ldots, p_n)$ is true in quantified Boolean logic iff $\mathcal{K}_{\mathcal{M}}, w_{\mathsf{G}} \models \phi$ where $\mathcal{K}_{\mathcal{M}}$ is the vision-based abstraction model of G. The idea encoded in that formula is that the knowledge operators $\mathsf{K}_{\mathsf{a}_i}$ and $\hat{\mathsf{K}}_{\mathsf{a}_i}$ serve respectively as universal and existential quantifiers over the truth values of the boolean variable p_i, while the guards $\mathsf{a}_j \triangleright \mathsf{a}_1$ ensure that the agents' cameras are positioned so as to enable each of these truth values. $\square$

COROLLARY 5.1. *The model checking problem for* $\mathsf{BBL_D}$ *is PSPACE-complete.*

5.3 Upper-bound for SAT(BBL_{PDL^*})

Now we address the problem of existence of a placement of cameras satisfying a given formula ϕ of BBL_{PDL^*}.

PROPOSITION 5.3. $SAT(\mathsf{BBL}^{PDL})$ *is in PSPACE.*

PROOF. The satisfiability testing procedure of a formula ϕ works as follows. First, we guess the set S_{a} for each agent a. Each S_{a} contains at most $O(k)$ sets of agents each of them containing at most $k = \#\mathsf{Agt}$ agents. Thus, it is possible to guess a collection of sets $(S_{\mathsf{a}})_{\mathsf{a} \in \mathsf{Agt}}$, where each S_{a} is a set of size at most $O(k^2)$ in polynomial time.

Now, we have to check that there exists a geometric model G from which we can derive the collection $(S_{\mathsf{a}})_{\mathsf{a} \in \mathsf{Agt}}$. That boils down to finding suitable positions and angles of vision for all cameras, consistent with the collection $(S_{\mathsf{a}})_{\mathsf{a} \in \mathsf{Agt}}$. The key point is that we can express the existence of such model in the existential fragment of the first-order theory of the field of reals (which is the same as the first-order theory of real-closed fields) [5].

Once we know the family $(S_{\mathsf{a}})_{\mathsf{a} \in \mathsf{Agt}}$ to be consistent, we compute the model $\mathcal{K}_{\mathcal{M}}$ induced by $(S_{\mathsf{a}})_{\mathsf{a} \in \mathsf{Agt}}$. We then select a set $\Gamma_{\mathsf{a}} \in S_a$ for each agent a. We finally check whether $\mathcal{K}_{\mathcal{M}}, (\Gamma_{\mathsf{a}})_{\mathsf{a} \in \mathsf{Agt}} \models \langle \overset{\frown}{\mathsf{a}_1} \rangle \ldots \langle \overset{\frown}{\mathsf{a}_n} \rangle \phi$ where $\mathsf{Agt} = \{\mathsf{a}_1, \ldots, \mathsf{a}_n\}$. For that purpose we call $mc((\Gamma_{\mathsf{a}})_{\mathsf{a} \in \mathsf{Agt}}, \langle \overset{\frown}{\mathsf{a}_1} \rangle \ldots \langle \overset{\frown}{\mathsf{a}_n} \rangle \phi)$ where the procedure mc is the procedure for the model checking. $\square$

5.4 Lower-bound for the satisfiability testing

PROPOSITION 5.4. $SAT(\mathsf{BBL}^{PDL})$ *is PSPACE-hard.*

PROOF. The formula $\exists p_1 \forall p_2 \ldots Q_n p_n \psi(p_1, \ldots, p_n)$ is true in QBL iff ϕ' is satisfiable where $\phi' = \langle \overset{\frown}{\mathsf{a}_1} \rangle [\overset{\frown}{\mathsf{a}_2}] \ldots O_i \psi'$ where O_i is $\langle \overset{\frown}{\mathsf{a}_i} \rangle$ if i is odd and $[\overset{\frown}{\mathsf{a}_i}]$ if i is even and ψ' is the formula ψ in which we have replaced all occurrences of p_i by $\mathsf{a}_i \triangleright \mathsf{a}_0$. $\square$

6. DETAILED ALGORITHMS

In this section, we provide the detailed pseudo-code of the algorithms for the decision procedures presented in the previous section.

6.1 Computing the vision-based abstract model

The algorithm first computes the sets S_a by calling function $ComputeVisionSets(\mathsf{G}, \mathsf{a})$ for each agent a, described in Figure 2, where $(x-y)[2\pi]$ denotes the angle of $[0, 2\pi)$ congruent with $x - y$; $T[0..N-1]$ is a circular array with indexes computed modulo the size of the array: $T[N] = T[0]$, $T[N+1] = T[1]$; and $\vec{u}$ is any unit vector of the plane. This preprocessing step runs deterministically in polynomial time and works as follows. $T.size$ denotes the number of elements of T, and $T[i].angle$ is an angle of $[0, 2\pi)$, $T[i].agents$ is the set of agents that are in direction determined by $T[i].angle$ from agent a. Usually $T[i].agents$ is a singleton except when several agents are aligned with a. Furthermore, T is sorted by angles. For instance, if $\mathsf{ang}(\mathsf{a}) = \frac{\pi}{2}$ and T is

	T[0]	T[1]	T[2]
$T[i].angle$	$\pi/6$	$\pi/3$	$11\pi/6$
$T[i].agents$	$\{b, c\}$	$\{d\}$	$\{e\}$

then $S_{\mathsf{a}} = \{\emptyset, \{\mathsf{b}, \mathsf{c}\}, \{\mathsf{b}, \mathsf{c}, \mathsf{d}\}, \{\mathsf{d}\}, \{\mathsf{e}\}, \{\mathsf{e}, \mathsf{b}, \mathsf{c}\}\}$. In the algorithm we simulate a complete turn of agent a. i_{begin} and i_{end} represents the first and the last index in T and they represents an interval of agents that may be seen by a. The corresponding set of agents is $agents(i_{begin}, i_{end}) = \bigcup_{i \in [i_{begin}, i_{end}]} T[i].agents$.

At $(*)$, we have computed the leftmost possible interval of agents that contains $T[0].agents$. Then we modify the current set of agents by adding the next agent of the list or by removing the last one if the next if too far. The computation of S_a is in $O(k \log k)$ where $k = \#\mathsf{Agt}$ (it is dominated by the cost of sorting).

6.2 Model checking the vision-based abstract model

In this section, we describe the pseudo-code of the procedures mc and $path?$ respectively in Figures 3 and 4. Note that in $path?$, $\mathbf{m}$ is written in binary in the expression $\gamma_1^{\mathbf{m}}$.

7. FURTHER EXTENSIONS OF THE BBL

The present work only considers a simplified and idealized scenarios of stationary cameras placed in the plane and without any obstacles to their vision. It is amenable to several natural and important extensions which we only briefly discuss here. The technical details and respective results will be presented in an extended follow-up work.

```
function ComputeVisionSets($\mathsf{G}, \mathsf{a}$)
    $S_\mathsf{a} := \emptyset$
    $T$ := circular array obtained by sorting the agents $\mathsf{b} \neq \mathsf{a}$ by
    increasing angle between vectors $\vec{u}$ and $\overrightarrow{(\mathsf{pos}(\mathsf{a}), \mathsf{pos}(\mathsf{b}))}$;
    if $T$ is empty then return $\{\emptyset\}$;
    $i_{begin} := 1$;
    $i_{end} := 0$;
    while $(T[i_{end}+1].angle - T[i_{begin}-1].angle)[2\pi] \leq \mathsf{ang}(\mathsf{a})$ do
        $i_{end} := i_{end} + 1$;
    endWhile
    $S_\mathsf{a}.add(agents(i_{begin}, i_{end}))$; (*)
    for $i := 1$ to $2 * T.size - 1$ do
        if $(T[i_{end}+1].angle - T[i_{begin}].angle)[2\pi] > \mathsf{ang}(\mathsf{a})$ then
            $i_{begin} := i_{begin} + 1$;
        else
            $i_{end} := i_{end} + 1$;
        endIf
        $S_\mathsf{a}.add(agents(i_{begin}, i_{end}))$;
    endFor
    return $S_\mathsf{a}$;
endFunction
```

Figure 2: An algorithm for computing S_a

```
function $mc(\Gamma, \phi)$
    match $\phi$ do
        case $\mathsf{a} \triangleright \mathsf{b}$: return $(\mathsf{b} \in \Gamma_\mathsf{a})$
        case $\neg\psi$: return not $mc(\Gamma, \psi)$
        case $\psi_1 \vee \psi_2$: return $mc(\Gamma, \psi_1)$ or $mc(\Gamma, \psi_2)$
        case $[\gamma]\psi$ :
            for $\Gamma' \in W$ do
                if $path?(\Gamma, \Gamma', \gamma)$ and not $mc(\Gamma', \psi)$ then
                    return false
                endIf
            endFor
            return true
    endMatch
endFunction
```

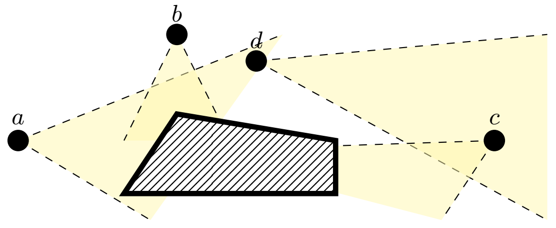
Figure 3: Algorithm for model checking for BBL_{PDL^*}

```
function $path?(\Gamma, \Gamma', \gamma)$
    match $\gamma$ do
        case $\overset{\frown}{\mathsf{a}}$ : return $(\Gamma'_\mathsf{b} = \Gamma_\mathsf{b}$ for all $\mathsf{b} \neq \mathsf{a})$
        case $\gamma_1; \gamma_2$ :
            for $\Gamma'' \in W$ do
                if $path?(\Gamma, \Gamma'', \gamma_1)$ and $path?(\Gamma'', \Gamma', \gamma_2)$
                    then return true
            endFor
            return false
        case $\gamma_1 \cup \gamma_2$: return $path?(\Gamma, \Gamma', \gamma_1)$or $path?(\Gamma, \Gamma', \gamma_2)$
        case $\phi?$: return $(\Gamma = \Gamma'$ and $mc(\Gamma, \phi))$;
        case $\gamma_1^*$ :
            if $\Gamma = \Gamma'$ then return true;
            for $\mathbf{m} := 1$ to $2^{(\#\mathsf{Agt})^2}$ do
                if $path?(\Gamma, \Gamma', \gamma_1^\mathbf{m})$ then return true;
            endFor
            return false;
        case $\gamma_1^\mathbf{m}$ with $\mathbf{m} = 1$: return $path?(\Gamma, \Gamma', \gamma_1)$;
        case $\gamma_1^\mathbf{m}$ with $\mathbf{m} > 1$ :
            for $\Gamma'' \in W$ do
                if $path?(\Gamma, \Gamma'', \gamma_1^{\lfloor \frac{\mathbf{m}}{2} \rfloor})$ and $path?(\Gamma'', \Gamma', \gamma_1^{\lceil \frac{\mathbf{m}}{2} \rceil})$
                    then return true
            endFor
            return false
    endMatch
endFunction
```

Figure 4: Algorithm for checking the existence of a γ-path from Γ to Γ'

7.1 Adding obstacles

In many real life scenarios the visibility by the cameras of the objects of interest, including other agents, is impaired by stationary or moving obstacles. Here we only sketch how our framework can be extended by adding stationary obstacles. In geometric models, these are represented by their shapes placed in the plane, as in the figure below.



In order to take obstacles into account, we have to modify the scan used to compute the sets S_a. For instance, In the example above, when we compute S_a, the segment $[\mathsf{pos}(\mathsf{a}), \mathsf{pos}(\mathsf{c})]$ intersects the obstacle and thus c will not appear in S_a.

Suppose that each obstacle is a polygon, described by a list of its vertices. A segment $[\mathsf{pos}(\mathsf{a}), \mathsf{pos}(\mathsf{b})]$ intersects the obstacle iff the segment intersects one of the segments of the polygon. Thus, testing intersection is done in polynomial time. Eventually, the model checking procedure can be adjusted to deal with obstacles and would remain in PSPACE.

The satisfiability testing problem now is modified as follows: given a set of obstacles positioned in the plane and a specification in BBL or any of its extensions, determine whether there is a positioning of the cameras in the plane (without moving the obstacles) such that the resulting configuration satisfies the specification. Its optimal complexity is currently under investigation.

7.2 Cameras in 3D

In reality, cameras are usually placed in the 3D space and their areas of vision are 3-dimensional. That adds essential, though not crucial, technical overhead in computing the areas of vision of the cameras, but the logical languages and the epistemic parts of their semantics remain essentially unchanged. The model checking and satisfiability testing algorithms, without or with obstacles, can be suitably modified and will be in the same complexity classes as in 2D.

7.3 Moving cameras and robots

In reality the agents and their cameras are often mobile, not stationary. The simplest way to capture that feature is to consider the same logical languages over geometric models that only involve the agents and the vision angles of their cameras, but not their positions. Thus, the model checking problem for the case of mobile cameras, asking for suitable positioning of the cameras, satisfying the input specification, reduces to satisfiability testing problem for models with stationary cameras. A more refined approach for capturing mobile cameras involves adding to the language an extra dynamic operator for translations in the plane along the current direction of the camera. Together with the operator for rotations (turning) these can express every movement in the plane. The respective geometric models involve again only the sets of agents and the vision angles of their cameras, but not their positions. The possible worlds (states) in Kripke models and in their vision-based abstraction models remain the same. They are based on all sets of other agents that may be seen in a glance after any rotation and/or translation. In the recursive model checking procedure for the enriched languages, every occurrence of a translation operator in the formula applied to an agent a would generate a finite number of different model checking problems, arising by sliding the camera in the current direction of the agent a and only recording the changes occurring in the vision sets of the agents. Thus, eventually, the model checking problem for the case of mobile cameras can be reduced to repeated solving of finite sets of model checking problem for models with stationary cameras. The optimal complexity of the model checking problem with moving cameras is currently open.

7.4 Communicating agents with cameras

Lastly, the agents controlling cameras may have the ability of communicating with each other so that their respective knowledge can be broadcast to others. The interaction between knowledge and announcements has been studied and it is known that latter do not have negative effect on the complexity of the basic logic (see [10]). We conjecture that in our case, too, adding announcements would not increase the complexity of the model checking and satisfiability.

8. CONCLUDING REMARKS

We have introduced formal models for multi-agent systems, where agents control surveillance cameras, and logical languages to reason about the interaction between their vision and knowledge. The abstract scenario considered in this paper is applicable in principle to various real-life situations, ranging from systems of real-time surveillance cameras in building and public areas, through multi-robot systems, to networks of stationary satellites positioned around the globe. Thus, our framework and results have immediate potential applications to automated reasoning, formal specification and verification of observational abilities and knowledge of multi-agent systems in each of these situations.

Acknowledgements

The work of Valentin Goranko was partially supported by the CIMI (Centre International de Mathematiques et d'Informatique) Excellence program while in Toulouse as a scientific expert.

9. REFERENCES

[1] P. Balbiani, O. Gasquet, and F. Schwarzentruber. Agents that look at one another. *Logic Journal of IGPL*, 21(3):438–467, 2013.

[2] M. Ben-Or, D. Kozen, and J. Reif. The complexity of elementary algebra and geometry. *Journal of Computer and System Sciences*, 32(2):251–264, 1986.

[3] P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic.* Cambridge University Press, 2001.

[4] A. L. Bustamante, J. M. Molina, and M. A. Patricio. Multi-camera and multi-modal sensor fusion, an architecture overview. In *Proc. of DCAI'2010*, pages 301–308, 2010.

[5] J. Canny. Some algebraic and geometric computations in PSPACE. In *Proceedings of STOC'88*, pages 460–467. ACM, 1988.

[6] R. Fagin, J. Halpern, Y. Moses, and M. Vardi. *Reasoning about Knowledge.* MIT Press, 1995.

[7] J. García, J. Carbó, and J. M. Molina. Agent-based coordination of cameras. *Intern. Journal of Computer Science & Applications*, 2(1):33–37, 2005.

[8] O. Gasquet, V. Goranko, and F. Scharzentruber. Big brother logic: Logical modeling and reasoning about agents equipped with surveillance cameras in the plane. In *Proc. of AAMAS'2014*, page ??, 2014.

[9] D. Harel, D. Kozen, and J. Tiuryn. *Dynamic Logic.* MIT Press, 2000.

[10] C. Lutz. Complexity and succinctness of public announcement logic. In *Proc. of AAMAS'06*, pages 137–143. ACM, 2006.

[11] F. Schwarzentruber. Seeing, knowledge and common knowledge. In *Logic, Rationality, and Interaction*, pages 258–271. Springer, 2011.

[12] M. Sipser. *Introduction to the Theory of Computation*, volume 2. Thomson Course Technology Boston, 2006.

[13] A. Tarski. *A decision method for elementary algebra and geometry.* Springer, 1951.

[14] H. van Ditmarsch, W. van der Hoek, and B. Kooi. *Dynamic Epistemic Logic.* Springer, Dordecht, 2008.