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

Olivier Gasquet  Valentin Goranko  François Schwarzentruber

AAMAS 2014

7 may 2014
Outline

1. Introduction
2. State of the art: robots
3. Contribution: cameras
4. Conclusion
Attack of a building

Aim
Represent the knowledge of agents.
Assumptions

It is common knowledge that:

- Agents are in the plane;
- Agents are transparent points;
- Agents see infinite cones;
- Angles for cones are the same;
- No obstacles;
Syntax

Epistemic modal logic with grounded atomic propositions

\[ \varphi, \psi, \ldots ::= \text{a sees b} \mid \neg \varphi \mid \varphi \lor \psi \mid K_a \varphi \mid CK_J \varphi \]

- \text{a sees b}: agent a sees agent b;
- \text{K}_a \varphi: agent a knows \varphi;
- \text{CK}_J \varphi: group J has common knowledge that \varphi.

Macro: \text{a sees b}: agent a does not see agent b.
Two settings for this epistemic logic

Robots (‘Flatland logic’)

- [Balbiani et al. IGPL. 2012]
- Agents that turn and move
  - Open issue: optimal algorithm for model checking and SAT

Cameras (‘Big brother logic’)

- [AAMAS 2014, TODAY]
- Common knowledge of the positions
  - Contribution: optimal algorithms for model checking and SAT.
Outline

1. Introduction

2. State of the art: robots
   - Semantics
   - Model checking

3. Contribution: cameras

4. Conclusion
Outline

1. Introduction

2. State of the art: robots
   - Semantics
   - Model checking

3. Contribution: cameras

4. Conclusion
Semantics: let us go for a Kripke model $\mathcal{M}_{\text{robots}}$

Definition (a possible world)

$$w = (pos, dir) \text{ where}$$

- $pos : AGT \rightarrow \mathbb{R}^2$
- $dir : AGT \rightarrow \{x \in \mathbb{R}^2 | ||x|| = 1\}$
Cone of vision
Semantics: epistemic relation in $M_{\text{robots}}$

**Epistemic relation**

$$w \sim_a u \text{ iff agent } a \text{ sees the same thing in both } w \text{ and } u.$$
Semantics: epistemic relation in $\mathcal{M}_{\text{robots}}$

**Epistemic relation**

$w \sim_a u$ iff agent $a$ sees the **same thing** in both $w$ and $u$. 

---

$\sim_a$

- **Agent $a$** sees the **same thing** between states $w$ and $u$. 
  - $a$ sees $c$, $d$, and $e$ in both $w$ and $u$. 
  - $a$ does not see $b$ in both $w$ and $u$. 

- **Agent $a$** does **not** see the **same thing** between states $w$ and $u$. 
  - $a$ sees $c$, $d$, and $b$ in $w$. 
  - $a$ sees $c$, $b$, and $e$ in $u$. 

Semantics: epistemic relation in $\mathcal{M}_{\text{robots}}$

Epistemic relation

$w \sim_a u$ iff agent $a$ sees the same thing in both $w$ and $u$. 

\[ \sim_a \]
Semantics: (artistic!) picture of $\mathcal{M}_{\text{robots}}$
Semantics: (artistic!) picture of $\mathcal{M}_{\text{robots}}$
Truth conditions

- $\mathcal{M}_{\text{robots}}, w \models (a \text{ sees } b)$ iff in $w$, $b$ is in the cone of vision of $a$;

- $\mathcal{M}_{\text{robots}}, w \models K_a \varphi$ iff for all $v \sim_a w$, $\mathcal{M}_{\text{robots}}, v \models \varphi$. 

\[ M_{\text{robots}}, w \models (a \text{ sees } b) \iff \text{in } w, \text{ } b \text{ is in the cone of vision of } a; \]

\[ M_{\text{robots}}, w \models K_a \varphi \iff \text{for all } v \sim_a w, M_{\text{robots}}, v \models \varphi. \]
Outline

1. Introduction

2. State of the art: robots
   - Semantics
   - Model checking

3. Contribution: cameras

4. Conclusion
Model checking

Input:
- $w = (pos, dir)$;
  (and certainly not the INFINITE Kripke model!);
- an epistemic formula $\varphi$.

Output:
- yes iff $\mathcal{M}_{\text{robots}}, w \models \varphi$.

Theorem

*The model checking problem is between PSPACE and EXPSPACE.*
*(reduction to $\mathbb{R}$-FO-SAT)*
Outline

1. Introduction
2. State of the art: robots
3. Contribution: cameras
   - Semantics
   - Abstraction works!
   - A PDL variant for cameras
   - Model checking
4. Conclusion
Outline

1. Introduction
2. State of the art: robots
3. Contribution: cameras
   - Semantics
   - Abstraction works!
   - A PDL variant for cameras
   - Model checking
4. Conclusion
Semantics: restricted set of worlds

**Assumption**

Common knowledge of the positions.

**Set of worlds**

Given a fixed $pos' : AGENTS \rightarrow \mathbb{R}^2$, worlds are $w = (pos, dir)$ s. th. $pos = pos'$
Semantics: restricted set of worlds

Assumption

Common knowledge of the positions.

Set of worlds

Given a fixed $pos' : AGENTS \rightarrow \mathbb{R}^2$,
worlds are $w = (pos, dir)$ s. th. $pos = pos'$
**Semantics: restricted set of worlds**

**Assumption**

Common knowledge of the positions.

**Set of worlds**

Given a fixed $pos' : AGENTS \rightarrow \mathbb{R}^2$, worlds are $w = (pos, dir)$ s. th. $pos = pos'$
Semantics: restricted set of worlds

**Assumption**

Common knowledge of the positions.

**Set of worlds**

Given a fixed $pos' : AGENTS \rightarrow \mathbb{R}^2$, worlds are $w = (pos, dir)$ s. th. $pos = pos'$
Semantics: $\mathcal{M}_{\text{cameras}}^{\text{pos'}}$

**Definition**

$\mathcal{M}_{\text{cameras}}^{\text{pos'}}$ is $\mathcal{M}_{\text{robots}}$ where we publicly announced the current positions $\text{pos}'$ of the agents.
Semantics: $\mathcal{M}^{pos'}_{cameras}$

Definition

$\mathcal{M}^{pos'}_{cameras}$ is $\mathcal{M}_{robots}$ where we publicly announced the current positions $pos'$ of the agents.
Semantics: $M^\text{pos'}_{\text{cameras}}$

**Definition**

$M^\text{pos'}_{\text{cameras}}$ is $M_{\text{robots}}$ where we publicly announced the current positions $\text{pos'}$ of the agents.
Outline

1. Introduction

2. State of the art: robots

3. Contribution: cameras
   - Semantics
   - Abstraction works!
   - A PDL variant for cameras
   - Model checking

4. Conclusion
Abstraction of the Kripke model $\mathcal{M}_{\text{cameras}}^{pos'}$

**Definition**

$$abs(w) = \{ b \text{ sees } c \mid \mathcal{M}_{\text{cameras}}^{pos'}, w \models b \text{ sees } c \}$$

**Theorem**

$abs$ is a bisimulation.

⚠️ This theorem is false for the robot settings.
**Compute the abstraction of** $\mathcal{M}_{\text{cameras}}^{\text{pos'}}$

**Abstract worlds are**

\[
\{ a \text{ sees } b \mid a \in \text{AGT} \text{ and } b \in \mathcal{V}_a \} \text{ where for all } a, \mathcal{V}_a \in \mathcal{V}_a.
\]

**Set of vision sets of agent $a$**

\[
\mathcal{V}_a = \{ \{ b \}, \emptyset, \{ c \}, \{ d \}, \{ d, f \}, \{ d, f, e \}, \{ f, e \}, \{ e \} \}.
\]

$\mathcal{V}_a$ computed in $O(\#\text{AGT} \log \#\text{AGT})$
Compute the abstraction of $M^{pos'}_{cameras}$

Abstract worlds are

\{ a \sees b \mid a \in AGT \text{ and } b \in v_a \} \text{ where for all } a, v_a \in V_a.

Set of vision sets of agent $a$

$V_a = \{ \{ b \}, \emptyset, \{ c \}, \{ d \}, \{ d, f \}, \{ d, f, e \}, \{ f, e \}, \{ e \} \}$.

$V_a$ computed in $O(\#AGT \log \#AGT)$.
Compute the abstraction of $M_{\text{cameras}}^{pos'}$

Abstract worlds are

\[ \{\text{a sees } b \mid a \in AGT \text{ and } b \in v_a\} \text{ where for all } a, v_a \in \mathcal{V}_a. \]

Set of vision sets of agent $a$

\[ \mathcal{V}_a = \{ \{b\}, \emptyset, \{c\}, \{d\}, \{d,f\}, \{d,f,e\}, \{f,e\}, \{e\} \}. \]

\[ \mathcal{V}_a \text{ computed in } O(#AGT \log #AGT) \]
Compute the abstraction of $M_{\text{cameras}}^{\text{pos'}}$

Abstract worlds are
\[ \{ a \text{ sees } b \mid a \in AGT \text{ and } b \in v_a \} \] where for all $a$, $v_a \in V_a$.

Set of vision sets of agent $a$
\[ V_a = \{ \{ b \}, \emptyset, \{ c \}, \{ d \}, \{ d, f \}, \{ d, f, e \}, \{ f, e \}, \{ e \} \}. \]

$V_a$ computed in $O(\#AGT \log \#AGT)$
Compute the abstraction of $M^{pos'}_{\text{cameras}}$

Abstract worlds are
\[
\{ a \text{ sees } b \mid a \in AGT \text{ and } b \in v_a \} \text{ where for all } a, v_a \in V_a.
\]

Set of vision sets of agent $a$
\[
V_a = \{ \{ b \}, \emptyset, \{ c \}, \{ d \}, \{ d, f \}, \{ d, f, e \}, \{ f, e \}, \{ e \} \}.
\]

$V_a$ computed in $O(#AGT \log #AGT)$
Compute the abstraction of $\mathcal{M}_{\text{cameras}}^{\text{pos'}}$

Abstract worlds are

\[ \{ a \text{ sees } b \mid a \in AGT \text{ and } b \in \nu_a \} \text{ where for all } a, \nu_a \in \mathcal{V}_a. \]

Set of vision sets of agent $a$

\[ \mathcal{V}_a = \{ \{ b \}, \emptyset, \{ c \}, \{ d \}, \{ d, f \}, \{ d, f, e \}, \{ f, e \}, \{ e \} \}. \]

$\mathcal{V}_a$ computed in $O(\#AGT \log \#AGT)$
Compute the abstraction of $M_{\text{cameras}}^{\text{pos}}$

Abstract worlds are

$$\{ a \text{ sees } b \mid a \in AGT \text{ and } b \in v_a \}$$

where for all $a$, $v_a \in V_a$.

Set of vision sets of agent $a$

$$V_a = \{ \{ b \}, \emptyset, \{ c \}, \{ d \}, \{ d, f \}, \{ d, f, e \}, \{ f, e \}, \{ e \} \}.$$

$V_a$ computed in $O(\#AGT \log \#AGT)$
Compute the abstraction of $M_{\text{cameras}}^{pos'}$ 

Abstract worlds are
$$\{ a \text{ sees } b \mid a \in AGT \text{ and } b \in v_a \}$$
where for all $a$, $v_a \in \mathcal{V}_a$.

Set of vision sets of agent $a$
$$\mathcal{V}_a = \{ \{ b \}, \emptyset, \{ c \}, \{ d \}, \{ d, f \}, \{ d, f, e \}, \{ f, e \}, \{ e \} \}.$$ 

$\mathcal{V}_a$ computed in $O(\#AGT \log \#AGT)$
Compute the abstraction of $M_{\text{cameras}}^{pos'}$

Abstract worlds are

$$\{ a \text{ sees } b \mid a \in AGT \text{ and } b \in \nu_a \}$$
where for all $a$, $\nu_a \in \mathcal{V}_a$.

Set of vision sets of agent $a$

$$\mathcal{V}_a = \{ \{ b \}, \emptyset, \{ c \}, \{ d \}, \{ d, f \}, \{ d, f, e \}, \{ f, e \}, \{ e \} \}.$$ 

$\mathcal{V}_a$ computed in $O(\#AGT \log \#AGT)$
Compute the abstraction of $M_{\text{cameras}}^{\text{pos}}$

Abstract worlds are
\[ \{ a \text{ sees } b \mid a \in AGT \text{ and } b \in v_a \} \text{ where for all } a, v_a \in V_a. \]

Set of vision sets of agent $a$
\[ V_a = \{ \{ b \}, \emptyset, \{ c \}, \{ d \}, \{ d, f \}, \{ d, f, e \}, \{ f, e \}, \{ e \} \}. \]

$V_a$ computed in $O(\#AGT \log \#AGT)$
Compute the abstraction of $M_{\text{cameras}}^{pos'}$

Abstract worlds are

$$\{ a \text{ sees } b \mid a \in AGT \text{ and } b \in v_a \} \text{ where for all } a, v_a \in V_a.$$ 

Set of vision sets of agent $a$

$$V_a = \{ \{ b \}, \emptyset, \{ c \}, \{ d \}, \{ d, f \}, \{ d, f, e \}, \{ f, e \}, \{ e \} \}.$$ 

$V_a$ computed in $O(\#AGT \log \#AGT)$
Compute the abstraction of $M_{\text{cameras}}^{pos'}$

Abstract worlds are

\[ \{ a \text{ sees } b \mid a \in AGT \text{ and } b \in v_a \} \text{ where for all } a, v_a \in V_a. \]

Set of vision sets of agent $a$

\[ V_a = \{ \{ b \}, \emptyset, \{ c \}, \{ d \}, \{ d, f \}, \{ d, f, e \}, \{ f, e \}, \{ e \} \}. \]

$V_a$ computed in $O(\#AGT \log \#AGT)$
Compute the abstraction of $M^{pos'}_{\text{cameras}}$

Abstract worlds are

$$\{a \text{ sees } b \mid a \in AGT \text{ and } b \in v_a\}$$

where for all $a$, $v_a \in V_a$.

Set of vision sets of agent $a$

$$V_a = \{ \{b\}, \emptyset, \{c\}, \{d\}, \{d,f\}, \{d,f,e\}, \{f,e\}, \{e\} \}.$$ 

$V_a$ computed in $O(\#AGT \log \#AGT)$
Compute the abstraction of $M^{pos'}_{cameras}$

Abstract worlds are

$$\{ a \text{ sees } b \mid a \in AGT \text{ and } b \in v_a \} \text{ where for all } a, v_a \in V_a.$$ 

Set of vision sets of agent $a$

$$V_a = \{ \{b\}, \emptyset, \{c\}, \{d\}, \{d,f\}, \{d,f,e\}, \{f,e\}, \{e\} \}.$$ 

$V_a$ computed in $O(\#AGT \log \#AGT)$
Compute the abstraction of $M_{\text{cameras}}^{\text{pos}}$

Abstract worlds are

$$\{ a \text{ sees } b \mid a \in AGT \text{ and } b \in \nu_a \} \text{ where for all } a, \nu_a \in \nu_a.$$  

Set of vision sets of agent $a$

$$\nu_a = \{ \{ b \}, \emptyset, \{ c \}, \{ d \}, \{ d, f \}, \{ d, f, e \}, \{ f, e \}, \{ e \} \}.$$  

$\nu_a$ computed in $O(#AGT \log #AGT)$
Compute the abstraction of $\mathcal{M}_{\text{cameras}}^{pos'}$\

Abstract worlds are\
\[
\{ a \text{ sees } b \mid a \in \text{AGT} \text{ and } b \in v_a \} \text{ where for all } a, v_a \in V_a.
\]

Set of vision sets of agent $a$\
\[
V_a = \{ \{b\}, \emptyset, \{c\}, \{d\}, \{d,f\}, \{d,f,e\}, \{f,e\}, \{e\} \}.
\]

$V_a$ computed in $O(\#\text{AGT} \log \#\text{AGT})$
Compute the abstraction of $M_{\text{cameras}}^{pos'}$

Abstract worlds are

\[ \{ a \text{ sees } b \mid a \in AGT \text{ and } b \in v_a \} \text{ where for all } a, v_a \in \mathcal{V}_a. \]

Set of vision sets of agent $a$

\[ \mathcal{V}_a = \{ \{ b \}, \emptyset, \{ c \}, \{ d \}, \{ d, f \}, \{ d, f, e \}, \{ f, e \}, \{ e \} \}. \]

\[ \mathcal{V}_a \text{ computed in } O(\#AGT \log \#AGT) \]
Compute the abstraction of $M_{\text{cameras}}^{\text{pos'}}$

Abstract worlds are

$$\{a \text{ sees } b \mid a \in AGT \text{ and } b \in v_a\}$$ where for all $a$, $v_a \in \mathcal{V}_a$.

Set of vision sets of agent $a$

$$\mathcal{V}_a = \{ \{b\}, \emptyset, \{c\}, \{d\}, \{d,f\}, \{d,f,e\}, \{f,e\}, \{e\} \}.$$ 

$\mathcal{V}_a$ computed in $O(\#AGT \log \#AGT)$
Compute the abstraction of $M_{\text{cameras}}^{pos'}$

Abstract worlds are

$$\{ a \text{ sees } b \mid a \in AGT \text{ and } b \in v_a \}$$ where for all $a$, $v_a \in \mathcal{V}_a$.

Set of vision sets of agent $a$

$$\mathcal{V}_a = \{ \{ b \}, \emptyset, \{ c \}, \{ d \}, \{ d, f \}, \{ d, f, e \}, \{ f, e \}, \{ e \} \}.$$
Compute the abstraction of $\mathcal{M}^{pos'}_{\text{cameras}}$

Abstract worlds are

$$\{a \text{ sees } b \mid a \in AGT \text{ and } b \in v_a\}$$

where for all $a$, $v_a \in \mathcal{V}_a$.

Set of vision sets of agent $a$

$$\mathcal{V}_a = \{\{b\}, \emptyset, \{c\}, \{d\}, \{d, f\}, \{d, f, e\}, \{f, e\}, \{e\}\}.$$

$\mathcal{V}_a$ computed in $O(\#AGT \log \#AGT)$
Outline

1 Introduction

2 State of the art: robots

3 Contribution: cameras
   - Semantics
   - Abstraction works!
   - A PDL variant for cameras
   - Model checking

4 Conclusion
Propositional Dynamic Logic (PDL) Language

**Grammar for formulas**

\[ \varphi, \psi, \ldots ::= a \text{ sees } b \mid \neg \varphi \mid \varphi \lor \psi \mid [\gamma]\varphi \]

- \([\gamma]\varphi\): after all executions of program \(\gamma\), \(\varphi\) holds.

**Grammar for programs**

\[ \gamma, \gamma' \ldots ::= \overset{a}{\leftarrow} \mid \varphi? \mid \gamma;\gamma' \mid \gamma \cup \gamma' \mid \gamma^* \]

| \(\overset{a}{\leftarrow}\) | a non-det chooses an angle and turns |
| \(\varphi?\) | test whether \(\varphi\) holds |
Translating epistemic operators in programs

$K_a$ is simulated by:

$$\gamma_a = \left[ (a \text{ sees } b_1 ? \cup (a \text{ sees } b_1 ?; \overline{b_1})) ; \ldots ; (a \text{ sees } b_n ? \cup (a \text{ sees } b_n ?; \overline{b_n})) \right]$$
Translating epistemic operators in programs

$K_a$ is simulated by:

\[
\gamma_a = \left[ (a \text{ sees } b_1? \cup (a \text{ sees } b_1?; \overset{\sim}{b_1})) ; \ldots ; (a \text{ sees } b_n? \cup (a \text{ sees } b_n?; \overset{\sim}{b_n})) \right]
\]
Translating epistemic operators in programs

$K_a$ is simulated by:

$$\left[ \left( a \text{ sees } b_1 ? \cup (a \text{ sees } b_1 ?; b_1) \right); \ldots; \left( a \text{ sees } b_n ? \cup (a \text{ sees } b_n ?; b_n) \right) \right]$$

Common knowledge among $\{a_1, \ldots, a_k\}$ is simulated by:

$$\left[ (\gamma_{a_1}; \ldots; \gamma_{a_k})^* \right]$$
Translating epistemic operators in programs

$K_a$ is simulated by:

$$
\left[ (a \text{ sees } b_1? \cup (a \text{ sees } b_1?; \underset{\sim}{b_1})) ; \ldots ; (a \text{ sees } b_n? \cup (a \text{ sees } b_n?; \underset{\sim}{b_n})) \right]_{\gamma_a}
$$
Translating epistemic operators in programs

$K_a$ is simulated by:

$$\left[ a \text{ sees } b_1 \ ? \cup (a \text{ sees } b_1 \ ? ; b_1^\uparrow) \right] ; \ldots ; \left[ a \text{ sees } b_n \ ? \cup (a \text{ sees } b_n \ ? ; b_n^\uparrow) \right]$$

Common knowledge among \{a_1, \ldots, a_k\} is simulated by:

$$\left[ (\gamma_{a_1} ; \ldots ; \gamma_{a_k})^* \right]$$
Outline

1 Introduction
2 State of the art: robots
3 Contribution: cameras
   - Semantics
   - Abstraction works!
   - A PDL variant for cameras
   - Model checking
4 Conclusion
Model checking

Theorem

Model checking of PDL for cameras is PSPACE-complete.

- **PSPACE-hard**
  We polynomially reduce QBF-SAT to the model checking of PDL for cameras.

- **PSPACE**
  We design an algorithm running in polynomial space.
Algorithm for model checking

```
function mc(w^{abs}, \varphi)
    match \varphi do
        case a sees b: return (a sees b \in \Gamma)
        case \neg \psi: return not mc(w^{abs}, \psi)
        case \psi_1 \lor \psi_2: return mc(w^{abs}, \psi_1) \lor mc(w^{abs}, \psi_2)
        case [\gamma]\psi:
            for u^{abs} \in W^{abs} do
                if ispath?(w^{abs}, u^{abs}, \gamma) and not mc(u^{abs}, \psi) then
                    return false
                endIf
            endFor
        return true
    endMatch
endFunction
```
Function *ispath*?

```plaintext
function ispath?(w^{abs}, u^{abs}, \gamma)
    match \gamma do
        case \overset{\rightarrow}{a} : return (w^{abs} = u^{abs}) except for what a sees;
        case \gamma_1; \gamma_2 :
            for v^{abs} \in W^{abs} do
                if ispath?(w^{abs}, v^{abs}, \gamma_1) and ispath?(v^{abs}, u^{abs}, \gamma_2)
                then return true
            endFor
            return false
        case \gamma_1 \cup \gamma_2: ...
        case \gamma^*: ...
    endMatch
endFunction
```
For $\gamma^*$: divide and conquer

Is there a path from $w^{abs}$ to $u^{abs}$ by $\gamma^*$?

- is there a path from $w^{abs}$ to $u^{abs}$ by $\gamma^K$ for some $K \in \{0, \ldots, 2^{AGT^2}\}$?
For $\gamma'^K$: divide and conquer

Is there a $\gamma'^K$-path?

$w^{abs}$

$u^{abs}$
For $\gamma^K$: divide and conquer
For $\gamma'^K$: divide and conquer

- $w^{abs}$
- $u^{abs}$

is there a $\gamma'^{\lfloor \frac{K}{2} \rfloor}$-path?
Use model checking for solving the satisfiability problem

**Definition**

Satisfiability problem

- input: a PDL formula $\varphi$;
- output: yes iff we can find positions and directions such that $\varphi$ holds.

**Theorem**

The satisfiability problem in PDL for cameras is PSPACE-complete.

- Non-deterministically guess vision sets;
- Check whether the vision sets are geometrically consistent; reduction into $FO(\exists \mathbb{R}) - SAT$;
- Non-deterministically guess a world $w^{abs}$;
- Call $mc(w^{abs}, \varphi)$. 
Outline

1. Introduction
2. State of the art: robots
3. Contribution: cameras
4. Conclusion
   - Summary
   - Future work
   - Demonstration
Outline

1. Introduction
2. State of the art: robots
3. Contribution: cameras
4. Conclusion
   - Summary
   - Future work
   - Demonstration
Summary

State of the art: variant for robots

- Abstraction is an open issue
- Model checking and SAT between PSPACE and EXPSPACE.

Contribution: variant for cameras

- Abstraction works
- Reduction in a PDL variant;
- Model checking and SAT are PSPACE-complete.
Outline

1. Introduction
2. State of the art: robots
3. Contribution: cameras
4. Conclusion
   - Summary
   - Future work
   - Demonstration
Different sensors

Sensor features
- Views: angle, radar;
- Myopic agents;
- Degree of quality in the perception.

Knowledge about other sensor features
- agent $a$ is myopic but agent $b$ does not know it;
A richer environment

- 3D;
- Some agents are ‘robots’ (or humans) and some are ‘cameras’;
- Obstacles.

**Satisfiability problem**

- input: a specification $\varphi$;
- output: yes, iff we can settle agents and obstacles such that $\varphi$. 
Including time, actions, strategies etc.

Reason about an attack of a building

intruder1

intruder2
Outline

1. Introduction
2. State of the art: robots
3. Contribution: cameras
4. Conclusion
   - Summary
   - Future work
   - Demonstration
Demonstration

Muddy children, prisoners, etc.
Thank you!