16th ICCRTS

"Collective C2 in Multinational Civil-Military Operations"

Title:
A Situation Analysis Toolbox for Course of Action Evaluation

ICCRTS Topics (in order of preference):
Topic 8: Architectures, Technologies, and Tools
Topic 3: Information and Knowledge Exploration
Topic 7: Modeling and Simulation

Authors:
Patrick Maupin and Anne-Laure Jousselme
R & D Defence Canada - Valcartier
2459 Pie XI North
Quebec, QC Canada, G3J 1X5
Patrick.Maupin@drdc-rddc.gc.ca; Anne-Laure.Jousselme@drdc-rddc.gc.ca

Hans Wehn, Snezana Mitrovic-Minic, Jens Happe
MacDonald, Dettwiler and Associates (MDA)
13800 Commerce Parkway
Richmond, BC Canada, V6V 2J3
hw@mdacorporation.com; SMITROVICMINIC@mdacorporation.com;
JHAPPE@mdacorporation.com

Point of Contact:
Hans Wehn
MacDonald, Dettwiler and Associates (MDA)
13800 Commerce Parkway
Richmond, BC Canada, V6V 2J3
e-mail: hw@mdacorporation.com

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
JUN 2011		**00-00-2011 to 00-00-2011**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
A Situation Analysis Toolbox for Course of Action Evaluation	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Defence R&D Canada - Valcartier,2459 Pie XI North,Quebec, QC Canada, G3J 1X5,**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
Presented at the 16th International Command and Control Research and Technology Symposium (ICCRTS 2011), Qu?c City, Qu?c, Canada, June 21-23, 2011. U.S. Government or Federal Rights License.

14. ABSTRACT
When positioning a number of static or mobile agents (human or autonomous) into an area for the purpose of surveillance, the positioning and motion strategies assigned to these agents impact on the situation awareness gained. This paper presents a toolbox that evaluates motion and placement strategies in a realistic surveillance context, using the Interpreted Systems semantics. Its five components implement relevant theoretical concepts: simulation of neutral and opposing forces? behavior, geometrical discretization of the Area of Interest, state space generation, state space analysis and visualization.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**	**Same as Report (SAR)**	**35**	

A Situation Analysis Toolbox for Course of Action Evaluation

Patrick Maupin, Anne-Laure Jousselme

R & D Defence Canada - Valcartier
2459 Pie XI North
Quebec, QC Canada, G3J 1X5
Patrick.Maupin@drdc-rddc.gc.ca

Hans Wehn, Snezana Mitrovic-Minic, Jens Happe

MacDonald, Dettwiler and Associates (MDA)
13800 Commerce Parkway
Richmond, BC Canada, V6V 2J3
hwehn@mdacoporation.com

Abstract – *When positioning a number of static or mobile agents (human or autonomous) into an area for the purpose of surveillance, the positioning and motion strategies assigned to these agents impact on the situation awareness gained. This paper presents a toolbox that evaluates motion and placement strategies in a realistic surveillance context, using the Interpreted Systems semantics. Its five components implement relevant theoretical concepts: simulation of neutral and opposing forces' behavior, geometrical discretization of the Area of Interest, state space generation, state space analysis, and visualization.*

Keywords: Situation analysis, situation awareness, interpreted systems, visibility graphs, pursuit-evasion.

1 Introduction[1]

Situation Awareness is essential to conduct decision-making activities. It is about the perception of the elements in the environment, the comprehension of their meaning, and the projection of their status in the near future [2]. Situation Analysis (SA) is defined as a process, the examination of a situation, its elements, and their relations, to provide and maintain a product, *i.e.* , a state of situation awareness for the decision maker [3]. This follows a standard intuition about levels 2 and 3 of the standard JDL Information Fusion model where, according to Steinberg and Bowman [4] Situation Assessment (level 2) is the "*estimation and prediction of entity states on the basis of inferred relations among entities*" whereas Impact Assessment (level 3) "*is usually implemented as a prediction, drawing particular kinds of inferences from Level 2 associations*".

The situation analysis process has to evaluate with both knowledge and uncertainty. A formalization is necessary if one is interested in the reproducibility of results, complex space-time problem solving, and in a language to represent and reason about dynamic situations. In [5], we proposed a formal model for situation analysis based on the interpreted systems semantics [6], a framework in which knowledge, information and uncertainty can be represented, combined, managed, reduced, increased, and updated.

The general problem of decision support can be sparsed into two principal tasks: situation analysis and planning. These two tasks are not easily separable since the actions carried out according to the plan defined during the task of planning modify the state of the environment and of the agents. This modifies the information to which the agents have access (their own or collective epistemic state) and thus their situational awareness. Our approach to situation analysis considers that the situation is created by the execution of a joint protocol for the agents P in interaction with the environment's protocol P_e, the latter including a possible opponent. The interpreted systems semantics will be used as the single specification language for both planning and situation analysis [7].

In this paper, we present a Situation Analysis Toolbox (SAT), which implements the theoretical concepts put forth in our previous works [8, 5, 7]. The general idea behind this toolbox is to build situations and analyse them. We use the formalism of the non-cooperative dynamic game theory [9] to model the situation. In its most general form it consists in a game between two teams having opposite goals, the players jointly seeking to maximize (resp. to minimize) a function of distance between targets. Differential game theory of Isaacs [10], dating back to the 1960s, made it possible to model the pursuit-evasion game - a simplified version of the problem of rendez-vous [11]. Isaacs' theory can be used to model many military problems by combining the traditional game theory of von Neuman and Morgenstein [12] (used especially in economy), variations calculus, and theory of optimal control. The interest to use such a framework for modelling the situations to be further

[1]This paper is a modified version of what has been presented in [1]. Part of the description in Section 2.2 have been also used in a paper by Van der Meyden *et al.* submitted at the IJCAI 2011 conference.

analyzed is the existence of some analytically derivable optimal solutions, thus helping to measure the quality of solutions for simple cases. Another advantage of this theoretical framework is its great flexibility. Indeed, by slightly modifying the constraints of the basic game, a large range of situations can be easily modeled: collision avoidance, target tracking, or stowing.

SAT consists of five (5) components (sub-toolboxes): (1) the Discretization Toolbox, which allows an abstraction of a continuous environment into both visibility and navigation graphs, (2) the Behaviour Simulation Toolbox, which enriches the environment with containment probability maps, (3) the State Generation Toolbox, which builds a transition state system based on joint strategies of some agents constrained by the context defined in (1) and (2), (4) the State Searching Toolbox, which plays the role of a model checker and temporal logical formulas verification in the transition state system, and (5) the Visualization Toolbox, which renders states and graphs on screen.

The paper is organised as follows. Section 2 is dedicated to theoretical background including formal definition of a situation, situation awareness, and situation analysis framed into the interpreted system semantics, to basics of pursuit-evasion games. This background section, also describes a counter-smuggling scenario situated in Howe Sound (on Canada's West Coast, northwest of Vancouver) on which the SAT will be illustrated. Section 3, presents the Situation Analysis Toolbox and details of its five components together with their related theoretical concepts. Finally, Section 4 provides further potential use and applications of the SAT.

2 Theoretical Background

In the following Section, we will give formal definitions of situations, situation awareness and situation analysis as well as basics on visibility-based pursuit-evasion games that will be used to model the situations.

2.1 Interpreted Systems for Situation Analysis

The interpreted systems semantics has been introduced in [6] and proposed in [8, 5] as a formal language for situation analysis.

Let $\mathcal{A} = \{1, 2, 3, \ldots, n, e\}$ be a set of agents, where e is a special agent denoting the *environment*. Each agent is assumed to be in some *local state* l_i at a given time, encapsulating all the information the agent has access to. l_e encodes all the relevant information that is not encoded in the agents' local states. In particular, l_e can encode the objective state of the world, which is usually not attainable by the agents' perception and reasoning means. For SA applications, apart from the objective states of the world, l_e can contain maps of the environment, network information, or any other information describing the outside world. The agents' local states can encode partial or imperfect views of this outside world.

A *global state* s is an element of $S \subseteq L_1 \times \ldots \times L_n \times L_e$, where L_i is the set of all possible local states of agent i. A sequence of global states $s^1, s^2, \ldots$ is called *run* r over S and can be viewed as a function from time to global states. A *system* $\mathcal{R}$ is a set of runs, and (r, m) denotes a *point* in $\mathcal{R}$. If $r(m) = (l_1, \ldots l_n, l_e)$ is the global state at point (r, m) (the state of the system at time m in run r), then $r_e(m) = l_e$ and $r_i(m) = l_i$. A *round* m in run r takes place between time $m-1$ and time m.

Actions are the cause of changes in the system and are performed by the agents and the environment in rounds. Let ACT_i be the set of actions that can be performed by agent i, $i = 1, \ldots, n$ and let ACT_e be the set of actions performed by the environment. A *joint action* is an element of $ACT_e \times ACT_1 \times \ldots \times ACT_n$, *i.e.* a tuple $(\mathbf{a}_e, \mathbf{a}_1, \ldots, \mathbf{a}_n)$ of actions performed by the set of agents and the environment. Λ denotes the null action.

A *protocol* P_i for the agent i is a mapping from the set L_i of local states of the agent i to nonempty sets of actions in ACT_i, $i \in \mathcal{A}$. A *joint protocol* P is a tuple $(P_1, P_2, \ldots, P_n)$ consisting of the protocols of each of the agents i, $i = 1, \ldots, n$. Note that P_e, the protocol of the environment, is not included in the joint protocol, but it is rather a part of the context.

A *context* γ is a tuple (P_e, S_0, τ, Ψ) where P_e is a protocol for the environment, S_0 is a nonempty subset of S describing possible states of the system at the initiation of the protocol, τ is a transition function, and Ψ is an admissibility condition on runs. The environment's protocol P_e can be used to model the adversary's strategies or simply to model random errors or events in a given situation. A *transition function* τ assigns to each global state and each joint action the resulting global state obtained after performing the joint action. The transition function describes which actions can be performed from a given global state, as well as the effect of these actions. The *admissibility condition* Ψ on runs tells which ones are "legal". In practice, Ψ can be used to shrink down a large system, or to model fairness conditions. Formally, Ψ is a set of runs and $r \in \Psi$ if and only if r satisfies the condition Ψ. Note that the description of the behaviour of a system is contextual, *i.e.* , a joint protocol P is always described within a given context γ.

Let Φ be a set of primitive propositions, describing basic facts about the system. Formulas are built using the classical operators of propositional logic. The set of formulas is closed under $\neg$ and $\wedge$ (negation and conjunction). Hence, given two formulas ϕ and ψ, $\neg\phi$, $\phi \wedge \psi$ are also formulas. Let $\mathcal{L}(\Phi)$ denote the language of Φ, *i.e.* the set of well-formed formulas.

An *interpreted system* (IS) $\mathcal{I}$ consists of a pair $\langle \mathcal{R}, \pi \rangle$ where $\mathcal{R}$ is a system over a set S and π is an interpre-

tation for the propositions in Φ over S, which assigns truth values (either true or false) to the primitive propositions at the global states. Thus, for every $p \in \Phi$ and state $s \in S$, we have $\pi(s)(p) \in \{false; true\}$.

We define a situation in terms of a transition state system in which the arcs are labeled by joint actions and nodes by global states. Formally, let $\mathcal{A}$ be a set of $n+1$ agents including the environment, and let $\mathcal{I}$ be the interpreted system representing the joint protocol P of the n agents in the interpreted context (γ, π).

Definition 2.1 (Situation [5]) *A* situation *is the sub-system $\mathcal{I}_{(r,m)}$ of $\mathcal{I}$, that is the system representing P in the interpreted context $(\gamma_{(r,m)}, \pi)$ where $\gamma_{(r,m)} = ((r,m), P_e, \Psi, \tau)$.*

Situation analysis is the process by which the decision maker (or analyst) reaches a state of situation awareness which will then allow him (or her) to make decisions.

Definition 2.2 (Situation analysis [5]) Situation analysis *is the process of verifying properties of $\mathcal{I}$. If ϕ is a formula of $\mathcal{L}(\Phi)$ expressing such a property, then analysing the situation amounts to answering the question*

$$(\mathcal{I}, r, m) \vDash \phi \tag{1}$$

If ϕ is satisfied in $\mathcal{I}$ at (r,m), then the analyst of the system is said to be aware *of ϕ.*

If the verification algorithm (*e.g.* model checking) is sound (*i.e.* always gives correct answers), then the analyst has explicit knowledge of ϕ (see [5] for details).

Epistemic model checking is used as a technique for (1) studying the consequences of the joint execution of protocols on the mental states of the agents, (2) data mining allowing to analyze in a qualitative way the state transition systems thus obtained, and possibly (3) allowing to check the existence of certain equilibrium conditions of games in terms of epistemic group notions.

Awareness however, is not simply a special state of knowledge but also refers to a limited capacity of the agents. Considering the limited resources of agents, we proposed in [5] a definition of awareness inspired by [13] and [14]:

Definition 2.3 (Awareness [5]) *The local state of each agent i at point (r,m) includes both a local algorithm $\mathsf{Aw}_i = \mathsf{alg}_i(r,m)$ and local data $l_i = \mathsf{obs}_i(r,m)$. An agent i of $\mathcal{A}$ is* aware *of ϕ at a given point (r,m) in $\mathcal{I}$, which we denote by $A_i\phi$, iff it is able to compute the truth value of ϕ:*

$$(\mathcal{I}, r, m) \vDash A_i\phi \text{ iff } \mathsf{A}_i(\phi, l_i) \neq \text{“?”} \tag{2}$$

The interpretation of A_i is thus understood as capturing any constraints like time, memory, reasoning abilities, etc.

Definition 2.4 (Situation awareness [5]) *For an agent i of $\mathcal{A}$, the* situation awareness *at point (r,m) is the set of formulas of $\mathcal{L}(\Phi)$ about which the agent i is aware:*

$$\mathsf{Aw}_i(r,m) = \{\phi \in \mathcal{L}(\Phi) | (\mathcal{I}, r, m) \vDash A_i\phi\} \tag{3}$$

Situation awareness is thus defined in terms of states, *i.e.* , in terms of a set of points in the system. For a given agent, the situation awareness is provided by test evaluations on observations about the environment (the objective state of the world). Situation awareness is both an epistemic ability and a precondition for action [7]. This is a practical definition of knowledge.

2.2 Situations generated by motion and sensing strategies

The games studied in this paper are special cases of motion and sensing problems, which [15] groups into four categories: sensor assignment, sensor placement, exploration, and pursuit-evasion. These four problems can be seen as special cases of a more general problem, involving either single or multiple sensors, static or mobile sensors, a searched target that is either punctual or distributed across the environment. The examples illustrating the functionalities of the SAT toolbox deal primarily with the resolution of pursuit-evasion (PE) games and sensor placement (SP), although we believe that the problems of exploration and sensor assignment can also be studied without or with minor modifications to the existing toolbox.

From a formal point of view, the simplest version of the sensor placement problem can be seen as a motion and sensing problem involving multiple and static agents aiming at maximizing their sensing coverage of a given surface of interest. A joint objective for the modeler is most of the time is also to minimize the number of sensors used. On the other hand, in the simplest pursuit-evasion problem the aim will be for a pursuer to minimize the distance to an evader, and vice-versa.

Problem solving situations such as pursuit-evasion are quite common, and come in various flavors. Also, depending on the scientific community many synonyms are used to designate somewhat similar problems: art gallery problems in computational geometry [16, 17], graph searching in computer science [18, 19] and [20], rendezvous problems in operations research [21, 22] and [23], and differential games in control theory [24].

A rough classification of pursuit evasion problems is based of the following variables: whether the environment is continuous or discrete, the pursuit success criterion is of kind (being caught or not) or of degree (getting close to evader for more than n seconds), involving a single pursuer (evaders) or multiple pursuers (evaders). Other agent-related variables distinguishing the pursuit-evasion problems involve sensing (whether agents' visibility is limited or not, uncertain or not), knowledge (whether the agents are

aware of the environment's map, of their own or other agents' position, whether there is or not an evader or pursuer in the environment), communication (whether multiple pursuer can communicate and thus coordinate the actions), steering ability (unequal agility for pursuers and evaders), motion capability (locally constrained, or jumping to other positions instantly), memory (whether the agents remember their past moves).

Local states encode physical terrain positions and hence the possible states of the agents can be stacked up the map, allowing convenient illustrations (see for example Figure 4(b)).

2.3 Counter-smuggling Vignette

We illustrate the SAT on a counter-smuggling vignette shown in Figure 1. In this figure, several boats manned by smugglers (the hostile agents) operate in Howe Sound (close to Vancouver). They attempt to go from their hiding places to target locations where their illegal goods can be exchanged. These agents try to avoid obstacles such as islands, and avoid colliding with each other. They also try to keep a distance away from the neutral agents such as ferries and pleasure crafts to avoid being seen and perhaps cause suspicion. More

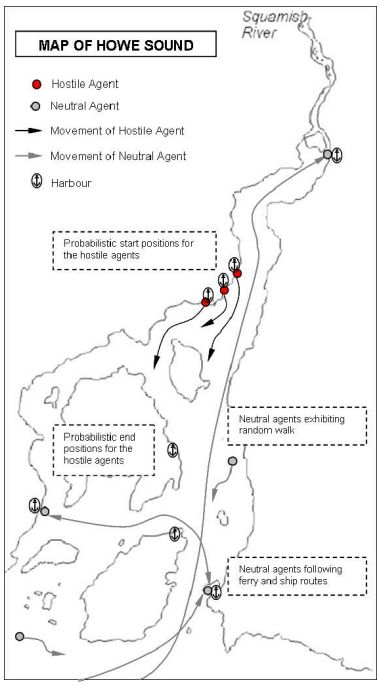


Figure 1: A smuggling operation has been reported in Howe Sound (north-west of Vancouver on Canada's West Coast). Can we guarantee that the smugglers will be detected ?

specifically, the neutral agents can be ferries or ships that follow a specific route, or boats that sail in any random direction. The locations of the home and target harbours are determined probabilistically.

3 Situation Analysis Toolbox

The SAT software is intended to provide tools for generating situations as well as analysing them. A key component of a situation is its context, which will be mainly represented here by the terrain on which the situation takes place. The area of interest is represented in the form of a polygon (*i.e.* boundary of the search area) with holes (which represent obstacles that the agents can neither see nor pass through). The software is divided into five sub-toolboxes, which serve different functions: (1) Discretization Toolbox, (2) State Generation Toolbox, (3) State Searching Toolbox, (4) Behaviour Simulation Toolbox and (5) Visualization Toolbox.

The high-level data flow diagram given in Figure 2 shows how these components are connected: First, the

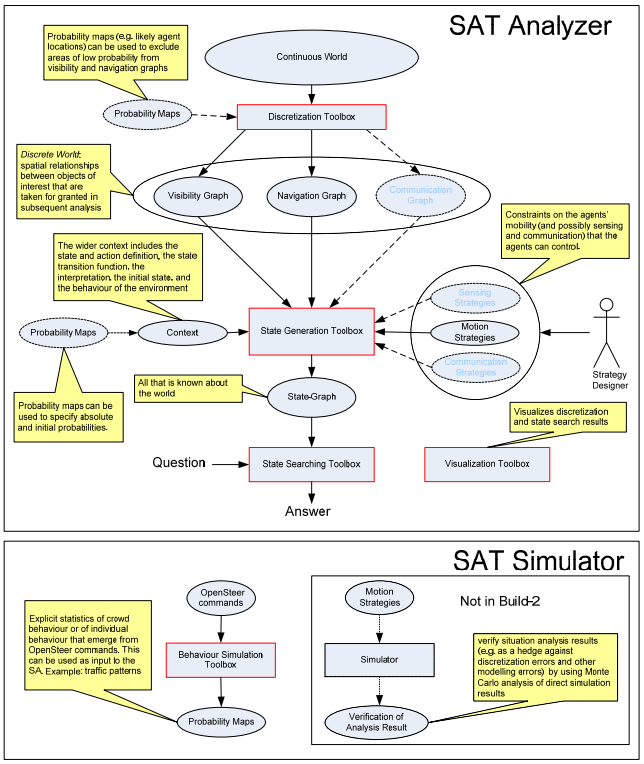


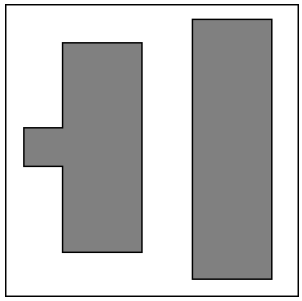
Figure 2: The Situation Analysis Toolbox.

Discretization Toolbox provides an abstraction of the terrain into a visibility graph and a navigation graph, which will then serve as the basis for defining a part of the context γ and the set of possible global states S for the system. Another component of the context comes from the Behaviour Simulation Toolbox, which provides agent containment probability maps of the area under consideration. The State Generation Toolbox inputs both the context $\gamma = (P_e, S_0, \tau, \Psi)$ and the strategies of the agents $P = [P_1, \ldots, P_n]$ and outputs a set of possible states. The situation thus built (*i.e.* , the transition state system) can then be analysed through the State Searching Toolbox, whose purpose is to answer logical queries of interest. The Visualization Toolbox provides

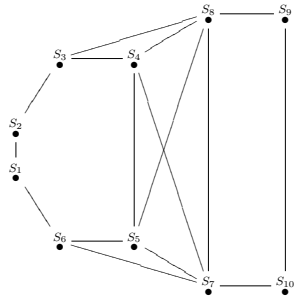
graphical user interface for the SAT.

3.1 The Discretization Toolbox

The main task of the Discretization Toolbox is to provide discrete representations of the real-world continuous search space geometry, suitable for situation analysis in the context of surveillance problems. The toolbox supports discrete representations in the form of either convex polygons (cells) as shown for instance in Figures 5(a) and 5(b), or points as shown in Fig. 3(b). Two main graphs are computed by the Discretization Toolbox: a visibility graph and a navigation graph.



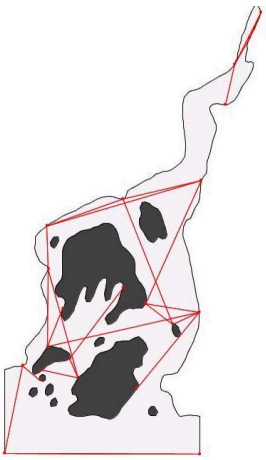
(a) Simple example of a polygonal environment with two holes (obstacles).



(b) Corresponding visibility graph over the set of reflex vertices.

Figure 3: Visibility graph computed for a polygonal environment.

(a) Map of Howe Sound.



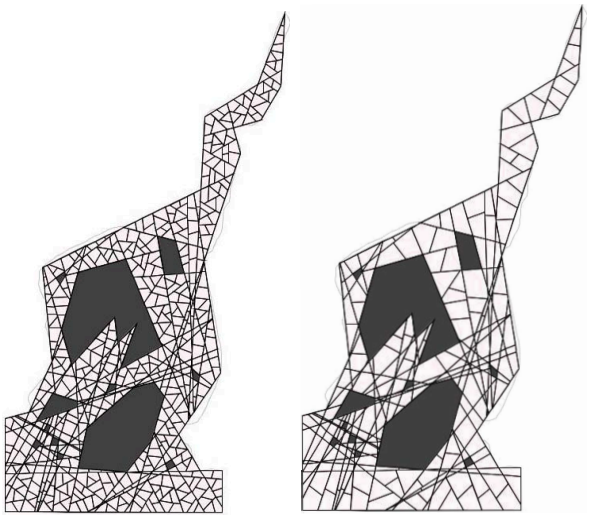
(b) 22 patrols (guards) with unlimited range sensors cover Howe Sound. Lines show visibility between guards.

Figure 4: GIS map and associated visibility graph.

The *visibility graph* of a set of points Q in a polygonal environment is a graph $\mathcal{G}_V(Q)$ whose nodes are the points in Q and in which two vertices are adjacent if they are visible from each other in the environment. An example is shown in Figure 3 for the reflex vertices in a simple polygonal environment, and in Figure 4 for the counter-smuggling scenario. The Discretization Toolbox provides several options for selecting the set Q from among the points in the polygonal environment. The *navigation graph* $\mathcal{G}_N$ defines the transition function τ for the agents. In an agent's navigation graph, the vertices are either points or cells, and two cells/points are connected if an agent can move directly from the one to the other in one time step. As a rule, the navigation graph is a subgraph of the visualization graph as two adjacent points/cells are always visible from one another. The agent's knowledge of the environment is built through an exploration algorithm whose aim is to learn the environment while exploring the navigation graph. Several algorithms such as the one described in [25] are available.

3.1.1 Sensor placement and different visibility ranges

Figures 6 show two sensor placement solutions together with the two corresponding discretizations of the environment. Figures 5(a) and 6(a) correspond to a visibility range of 1km, while Figures 5(b) and 6(b)



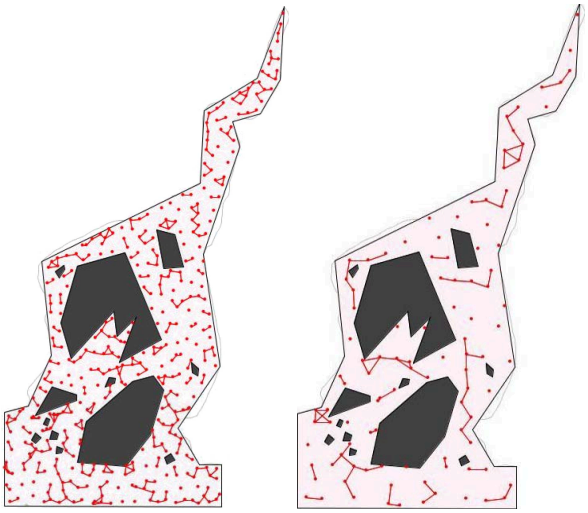
(a) Decomposition obtained for a visibility range of 1000 meters.

(b) Decomposition obtained for a visibility range of 2000 meters.

Figure 5: Cell decomposition of the environment for two visibility ranges.

correspond to a visibility range of 2km. In these figures, all the islands (obstacles) are retained in the map. The initial Howe Sound map has been simplified using a simplification parameter of 1000m, meaning that all the lines smaller than 1000m are removed. The simplification is used because the algorithm for positioning sensors with limited visibility range generates a search space decomposition as its initial step and decomposition can be computationally expensive.

Decreased sensor visibility range or deteriorating weather conditions may cause the increase in the number of sensors needed to monitor the area. Table 1 gives



(a) Sensor placement solution with visibility range of 1000 meters over a map simplified with a parameter of 1000 meters.

(b) Sensor placement solution with visibility range of 2000 meters over a map simplified with a parameter of 1000 meters.

Figure 6: Sensor placement solutions and corresponding visibility graphs for different visibility ranges.

indications on the number of sensors required when the visibility range changes from 5km to 1km. The sensors are positioned until the 100% coverage is achieved. The provided running times are obtained by running

Visibility range	Potential sensor locations	Sensor placement	Running time
Unlimited	75	13	0:03.14
5000m	386	27	0:16.92
4000m	389	33	0:25.68
3000m	395	57	3:05.70
2000m	444	114	5:12.23
1000m	774	378	1h 10:26.39

Table 1: Sensor placement for different visibility ranges.

the experiments on IBM ThinkPad, with Intel Core 2 Duo CPU T9600 at 2.80GHz, and 1.98GB RAM.

3.1.2 Sensor placement and critical locations for PE games

Different map sizes and corresponding number of sensors and critical locations for cooperative guards in a PE games are tested on the same map (Howe Sound) with different simplification parameters and the results are shown in Table 2. As above, the sensors are placed until reaching a coverage of 100%. The "Sensor placement" column represents the number of sensors that are required to cover the entire search polygon (100%) in the case of unlimited visibility range sensors (visibility is obstructed by obstacles (islands) only). For such a sensor placement, the art gallery algorithms have been used. The "PE game: critical loc." column represents the number of critical locations that needs to be visited by pursuer(s) in order to find an evader. These locations represent the locations of cooperative guards. Cooperative guards cover entire search polygon and the visibility graph whose nodes are cooperative guard locations is connected. The set of cooperative guards is used for the PE game because (i) the pursuers can move between the critical locations in straight lines and (ii) pursuers are able to visit all the locations (clear the entire search polygon) by walking along the edges of the visibility graph. We observe that the larger the simplification parameter is, the smaller number of vertices of the search space will be.

3.2 The State Generator

The State Generation Toolbox strives to be an implementation of the concepts and abstractions introduced in the first part of this paper. It incorporates the concepts of state, agent, action, transition function, and strategy. It provides a state graph which represents an interpreted system $\mathcal{I}$ for the purpose of testing agent behaviour strategies. This module takes as main inputs

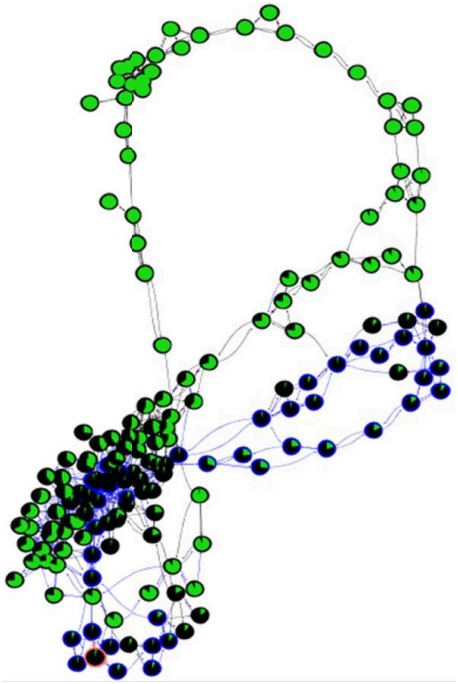


Figure 7: Display of the state space.

the abstraction of the terrain under the form of both the visibility and navigation graphs computed by the Discretization Toolbox. Other inputs are: the agents' local states definition l_i which may include other parameters than the positions, the agents's protocols P_i, the context $\gamma = (P_e, S_0, \tau, \Psi)$ and the interpretation function π which indicates whether some Boolean formulas ϕ are true in global states s. Given a state, the State

Table 2: Number of sensors and critical locations for a PE game with cooperative guards with different map simplifications.

Decomp. range (m)	Vertices	Reflex vertices	Sensor placement	Running time	PE games: Critical loc.	Running time
As is	537	316	22	6:15.57	23	8:57.42
10	473	284	22	3:48.36	24	5:58.87
50	333	212	20	1:35.26	22	2:59.12
100	253	166	18	0:33.52	20	1:17.26
250	145	99	15	0:09.14	17	0:12.65
500	97	71	14	0:05:27	15	0:06.99
1000	75	62	13	0:03.14	15	0:04.24
2000	67	57	12	0:02.74	12	0:03.64

Generation Toolbox outputs a set of successor states. Figure 7 is a display of the state space corresponding to a search in a PE game. Circles represent global states and are colored in green according the proportion of cleared cells: A black circle means that no state has been cleared while a full green circle means that all the states have been cleared. The red circle identifies the current state.

3.3 The State Searching Toolbox

The main task of the State Searching Toolbox is to provide answers to formal queries by searching the state graph $\mathcal{I}$. This module takes as input the state transition function τ and a query ϕ, and outputs the query result. Although basic answers to queries on state transition systems are Boolean (either true or false), degrees can also be computed ranging from probably true to probably false. Also, additional statistical measures can be computed such as: Number of states where ϕ is true/false, Number of paths where ϕ is true/false, Length of path where ϕ is true/false, Length of path before encountering a counterexample, Average length of path, Ratio of states/paths. The formal language of queries is CTL, which is propositional logic over the states in a state graph, augmented by the list of operators shown in Table 3.

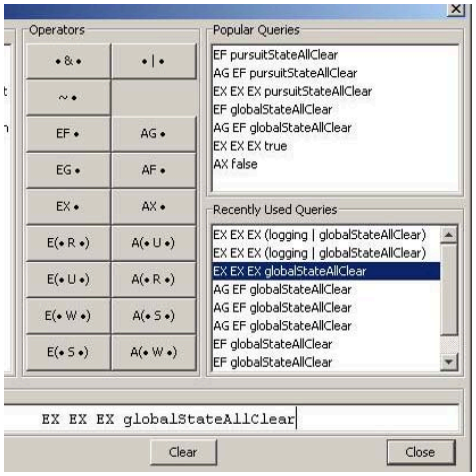


Figure 8: CTL query Graphical User Interface.

AX ϕ	ϕ in all next states.
EX ϕ	ϕ in at least one next state.
A $[\phi$ **U** $\psi]$	on all paths, ϕ until ψ.
E $[\phi$ **U** $\psi]$	on at least one path, ϕ until ψ.
AF ϕ	On all paths, in some future state, ϕ.
EF ϕ	On at least one path, in some future state, ϕ.
AG ϕ	On all paths, in all future states, ϕ.
EG ϕ	On at least one path, in all future states, ϕ.

Table 3: CTL operators.

The State Searching Toolbox plays the role of a model checker and thus serves as the situation analysis tool proper. Figure 9 shows an instance of the counter-smuggling vignette. The previously computed restricted search space is discretized into cells, of which the green cells are used to compute the visibility graph. In a first step, (both the purple and green) cells have been selected through a threshold on the probability map (see for example Fig. 11(b)). In a second step, a subset of the thresholded cells is further selected and colored green. The green cells are sufficient to cover all thresholded cells using infinite-range omnidirectional sensors. The green guy marks the start position of the pursuer, who is supposed to systematically search for evaders in the gallery. The predicate `pursuitStateAllClear` used in the query ascertains that (all cells in) a given state is (are) clear of evaders—or equivalently, that any evader present in the search area has been caught. The results of past CTL queries are shown on the right-hand-side of the screen. In this case, they return false because the topology of the area allows an evader to hide behind islands and avoid detection, no matter how the pursuer moves. Multiple pursuers would be necessary to completely sweep the area and guarantee detection.

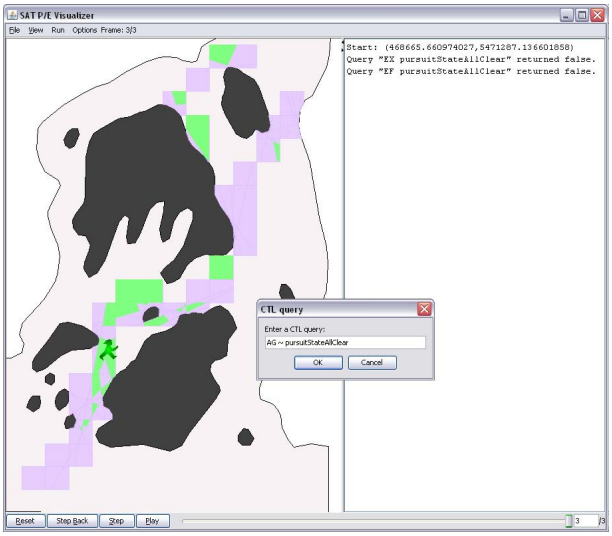


Figure 9: Executing queries using the Visualizer Toolbox for the counter-smuggling vignette.

3.4 The Behaviour Simulation Toolbox

The main task of the Behaviour Simulation Toolbox (see Figure 10) is to extract probability maps for agent containment from OpenSteer[2] multi-agent simulations. The OpenSteer is a library implementing "steering behaviours" originally designed by Craig Reynolds [26] to model coordinated motion of animals such as flying birds. Given the description of the scenario in Section

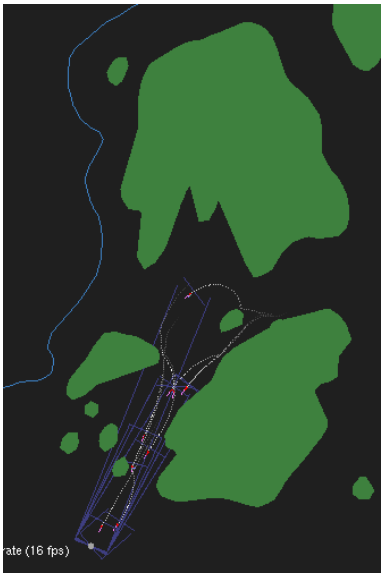
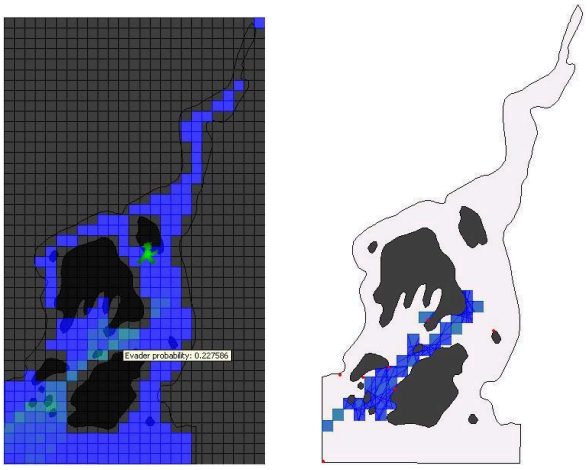


Figure 10: Screen shot of the Behaviour Simulation Toolbox for the counter-smuggling vignette.

2.3, the Behaviour Simulation Toolbox allows the generation of a map of the most likely locations for finding the smugglers. The main input to the Behaviour Simulation Toolbox is a GIS map corresponding to the area of interest. The output is a probability map, *i.e.* a set of polygonal cells marked up with their probability of agent containment (Fig. 11).

[2] `http://opensteer.sourceforge.net`



(a) Probability map of evader presence. Lighter, greenish colours represent higher probability.

(b) Restricted search space: 0.1 probability threshold.

Figure 11: Probability map output by the Behaviour Simulation Toolbox.

3.5 The Visualization Toolbox

The Visualization Toolbox allows the user to run and visualize the following: discretization of a map into either cells or points (guards) (Figure 4(b)), input of CTL queries for the state search, display of the results of the query, and visualization of the graph search (Figure 9).

The state generator and the visualizer currently support the following problem types [27]: the pursuit-and-evade problem, the sensor placement problem, and the exploration problem.

4 Conclusions

We presented a Situation Analysis Toolbox implementing formal notions of situation analysis based on epistemic transition state systems. Composed of 5 subtoolboxes, the SAT builds a situation based on an abstract version of the environment as a visibility graph and on the execution by a set of agents of a joint strategy derived from pursuit-evasion game theory. The context is further enriched with probability maps of presence of agents, built through modeling emerging behaviour. Using the situational definitions and formalisms, SAT affords modeling, evaluation, and representation of dynamic environments

Acknowledgments

The authors want to thank Grace Lin for programming on the behaviour simulation toolbox.

References

[1] Patrick Maupin, Anne-Laure Jousselme, Hans Wehn, Snezana Mitrovic-Minic, and Jens Happe. A situation analysis toolbox: Application to coastal and offshore surveillance. In *Proceedings of the 13th International conference on Information Fusion*, Edinburgh, Scotland, July 2010.

[2] Mica R. Endsley and Daniel J. Garland. *Situation Awareness Analysis and Measurement*. Lawrence Erlbaum Associates, Publishers, Mahwah, New Jersey, 2000.

[3] Jean Roy. From data fusion to situation analysis. In *Proceedings of the 4th International Conference on Information Fusion*, volume II, pages ThC2–3 – ThC2–10, Montreal, Canada, 2001.

[4] A. N. Steinberg and C. L. Bowman. Revisions to the JDL data fusion model. In D. L. Hall and J. Llinas, editors, *Handbook of Multisensor Data Fusion*, The Electrical Engineering and Applied Signal Processing Series, chapter 2, pages 2–1 to 2–19. CRC Press, Boca Raton, 2001.

[5] Patrick Maupin and Anne-Laure Jousselme. Interpreted systems for situation analysis. In *Proceedings of the 10th International Conference on Information Fusion*, Quebec City, Canada, 9-12 July 2007.

[6] Ronald Fagin, Joseph Y. Halpern, Yoram Moses, and Moshe Y. Vardi. *Reasoning about knowledge*. The MIT Press, Cambridge, MA, 2003.

[7] Anne-Laure Jousselme, Patrick Maupin, Christophe Garion, Laurence Cholvy, and Claire Saurel. Situation awareness and ability in coalitions. In *Proceedings of the 10th International Conference on Information Fusion*, Quebec City, Canada, 2007.

[8] Patrick Maupin and Anne-Laure Jousselme. A general algebraic framework for situation analysis. In *Proceedings of the 8th International Conference on Information Fusion*, Philadelphia, PA, USA, July 2005.

[9] Tamer Basar and Geert Jan Olsder. *Dynamic noncooperative game theory*, volume 160 of *Mathematics in Science and Engineering*. Academic Press, 1999.

[10] R. Isaacs. *Differential games*. Wiley, New York, 1965.

[11] Steve Alpern and Shumel Gal. *The Theory of Search Games and Rendezvous*. Kluwer Academic Publishers, 2002. 336 pages.

[12] J. von Neuman and O. Morgenstein. *Theory of Games and Economic Behaviour*. Princeton University Press, Princeton, 1944.

[13] Ronald Fagin and Joseph Y. Halpern. Belief, awareness, and limited reasoning. *Artificial Intelligence*, 34(1):39–76, 1988.

[14] J. Y. Halpern, Y. Moses, and M. Y. Vardi. Algorithmic knowledge. In *Proc. of the 5th Conference on Theoretical Aspects of Reasoning about Knowledge (TARK'94)*, pages 255–266. Morgan Kaufmann, 1994.

[15] I. Volkan Isler. *Algorithms for Distributed and Mobile Sensing*. PhD thesis, University of Pennsylvania, 2004.

[16] V. Chvàtal. A combinatorial theorem in plane geometry. *Journal of Combinatorial Theory Series B*, 18:39–44, 1975.

[17] I. Bjorling-Sachs and D. Souvaine. An efficient algorithm for guard placement in polygons with holes. *Discrete and Computational Geometry*, 13:77–109, 1995.

[18] N. Megiddo, S. L. Hakimi, M. R. Garey, D. S. Johnson, and C. H. Papadimitriou. The complexity of searching a graph. *Journal of the ACM*, 35(1):18–44, 1988.

[19] Arthur S. Goldstein and Edward M. Reingold. The complexity of pursuit on a graph. *Theoretical Computer Science*, 143:93–112, 1995.

[20] Shmuel Gal. Strategies for searching graphs. In *Graph Theory, Combinatorics, and Algorithms*, pages 189–214. Springer, 2005.

[21] W. S. Lim. A rendezvous-evasion game on discrete locations with joint randomization. *Advances in Applied Probability*, 29:1004–1017, 1997.

[22] S. Alpern and W. S. Lim. The symmetric rendezvous-evasion game. *SIAM Journal on Control and Optimization*, 36(3):948–959, 1998.

[23] Steve Alpern. Rendezvous search: A personnal perspective. *Operations Research*, 50(5):772–795, 2002.

[24] Rufus Isaacs. *Differential Games: A Mathematical Theory with Applications to Warfare and Pursuit, Control and Optimization*. Wiley and Sons (1965), Dover (1999), 1965.

[25] D. Pellier and H. Fiorino. Coordinated exploration of unknown labyrinthine environments applied to the pursuit evasion problem. In *Proceedings of the fourth international joint conference on Autonomous Agents and Multiagent Systems*, pages 895–902. ACM Press New York, NY, USA, 2005.

[26] Craig W. Reynolds. Steering behaviors for autonomous characters. In *Proc. of Game Developers Conference*, 1999.

[27] Ibrahim Volkan Isler. *Algorithms for Distributed and Mobile Sensing*. PhD thesis, University of Pennsylvania, 2004.

A Situation Analysis Toolbox for Course of Action Evaluation

Patrick Maupin, Anne-Laure Jousselme, Hans Wehn, Snezana Mitrovic-Minic, Jens Happe

R & D Defence Canada - Valcartier

MacDonald, Dettwiler and Associates (MDA)

Defence Research and Development Canada

Recherche et développement pour la défense Canada

Canada

Outline

1. Formalisation of the Situation Analysis process
   - Situation, Situation awareness, Situation analysis
2. Situation Analysis Toolbox (SAT) implementing the previous theoretical concepts
   - Modeling situation as a pursuit-evasion game
   - Counter-smuggling vignette
   - Five Modules
     1) *Behaviour simulation toolbox*
     2) *Discretisation toolbox*
     3) *State generation toolbox*
     4) *State searching toolbox*
     5) *Visualization toolbox*
3. Conclusions

Interpreted Systems Semantics for situation analysis

Interpreted systems semantics is an epistemic logical approach proposed in the 1995 for the analysis of distributed systems by **Fagin**, **Halpern**, **Moses** and **Vardi**

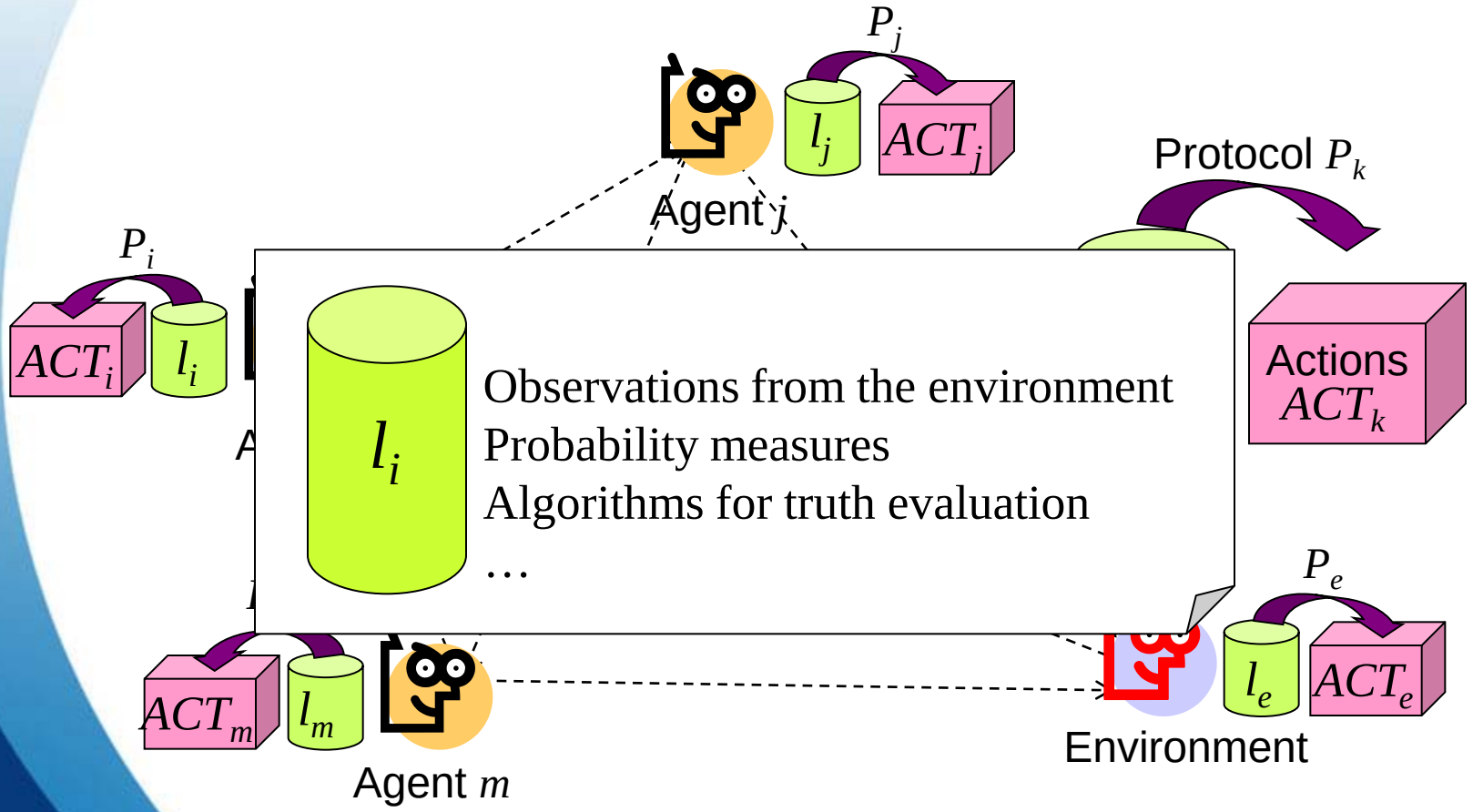
R. Fagin, J. Y. Halpern, Y. Moses, and M. Y. Vardi. Reasoning about knowledge. The MIT Press, Cambridge, MA, 2003.

Interpreted Systems Semantics for situation analysis

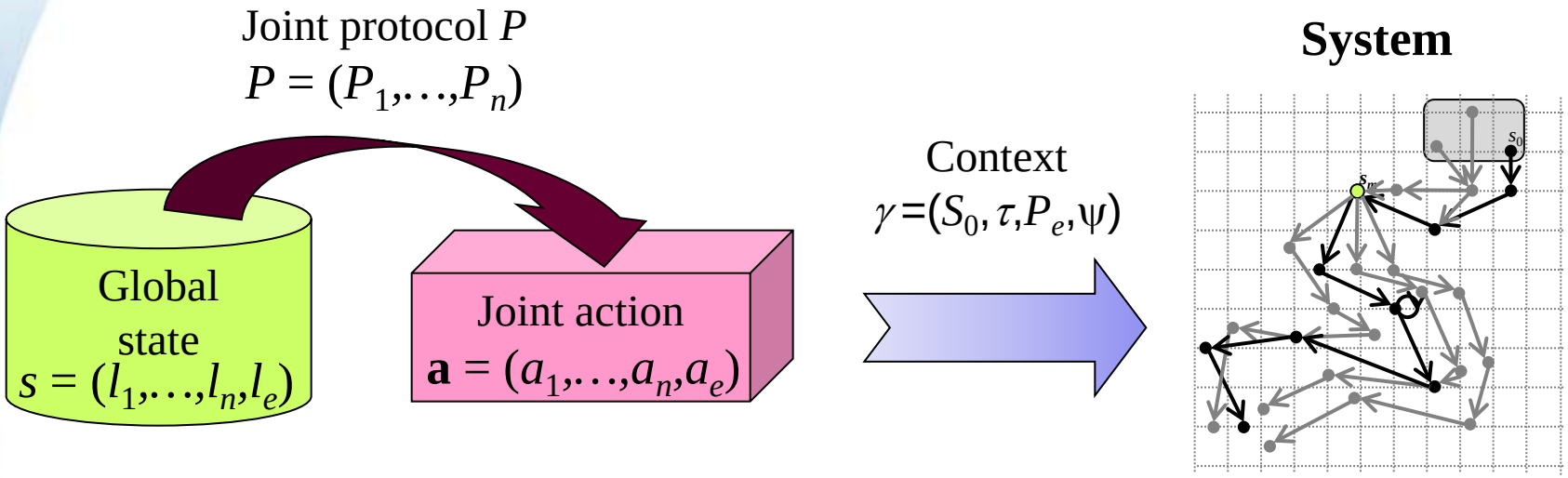
- Hypothesis: ***Interpreted Systems Semantics*** *is a general framework for* ***situation analysis*** *and high-level information fusion applications*
- Arguments:
  - Designed for **distributed systems** analysis;
  - **Situations** are adequately represented by state transition systems;
  - The notions of Situation, Situation Awareness and Situation Analysis can be **formally defined;**
  - Allows reasoning about **knowledge**, **uncertainty** and **time;**
  - The framework is general enough so that **Generalized Information Theory** can be **framed** into ISS;
  - Can take advantage of both **model checking** and inductive decision **procedures**.

P. Maupin and A.-L. Jousselme. A general algebraic framework for situation analysis. In *Proc. of the 8th Int. Conf. on Information Fusion, Philadelphia, PA,* USA, July 2005.

Elements of the model (1)



Elements of the model (2)



$$\mathcal{I} = \langle S, P, \gamma, \pi \rangle$$

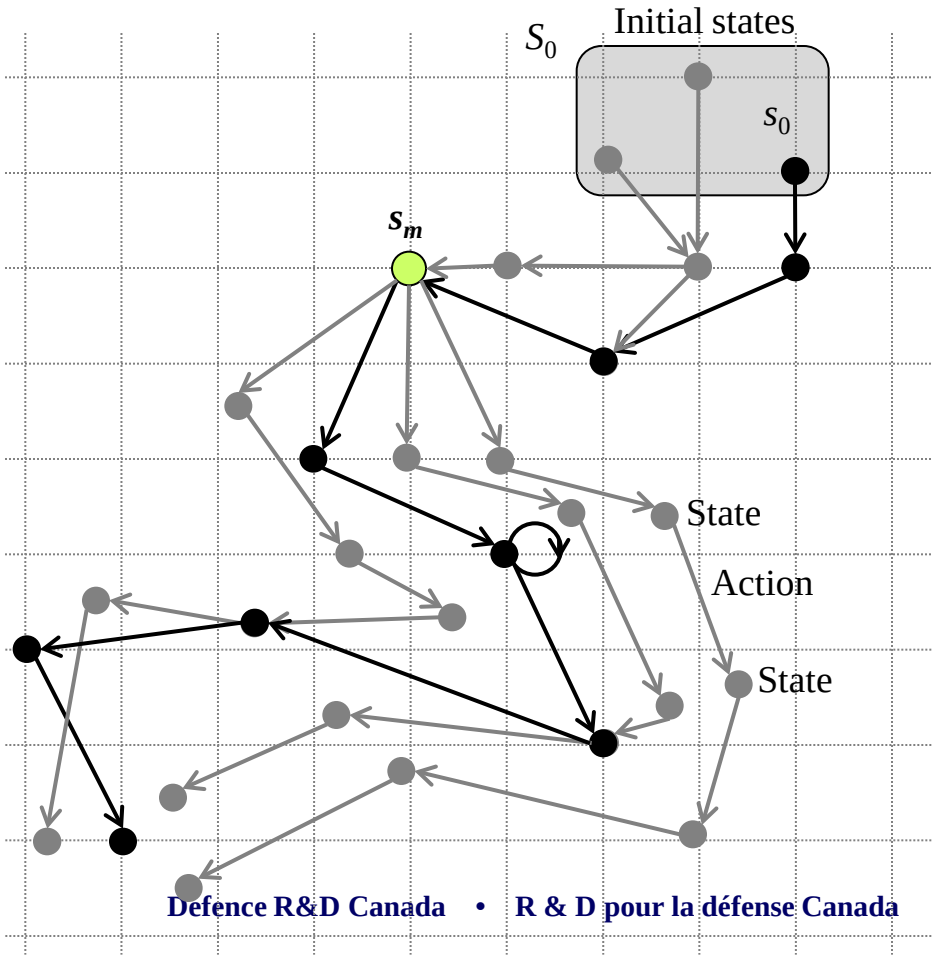
π is an interpretation function for formulas in $\mathcal{L}(\Phi)$

Situation

A *situation* is the subsystem $\mathcal{I}(r,m)$ of $\mathcal{I}$, that is the system representing P in the interpreted context $(\gamma_{(r,m)},\pi)$

<u>3 remarkable cases:</u>

1. Global state
2. Given a single initial state
3. Full spectrum of possible paths



Awareness as resource-boundedness

An agent is *aware* of a formula ϕ if it is able to compute its truth value

$$(\mathcal{I}, r, m) \vDash A_i\phi \text{ iff } \mathsf{A}_i(\phi, l_i) \neq \text{“?”}$$

Algorithm for truth evaluation

$$\mathsf{A}_i = \mathsf{alg}_i(r, m)$$

Local data = Observations

$$l_i = \mathsf{obs}_i(r, m)$$

with $\mathsf{A}_i(\phi, l_i)$ =
- Yes if ϕ is true
- No if ϕ is false
- ? if the agent is unable to compute

→The fact that the algorithm can compute the truth value of ϕ does not mean that this is the correct truth value.

→ Awareness is a practical notion of knowledge

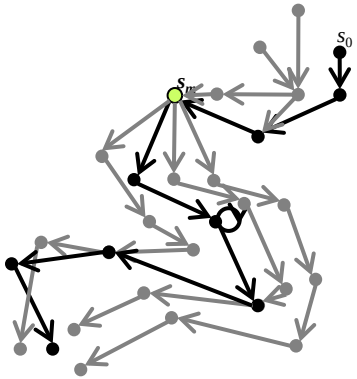
Situation analysis *(proposed approach)*

Situation analysis is the process of verifying properties of the interpreted system expressed by a formula ϕ_{KT}

$$(\mathcal{I}, r, m) \models \phi_{KT}$$

Queries

Is ϕ true ?
Will ϕ be true at all future steps ?
Is ϕ always true ?
Does Agent 1 knows ϕ ?
Does everybody in group G knows ϕ ?

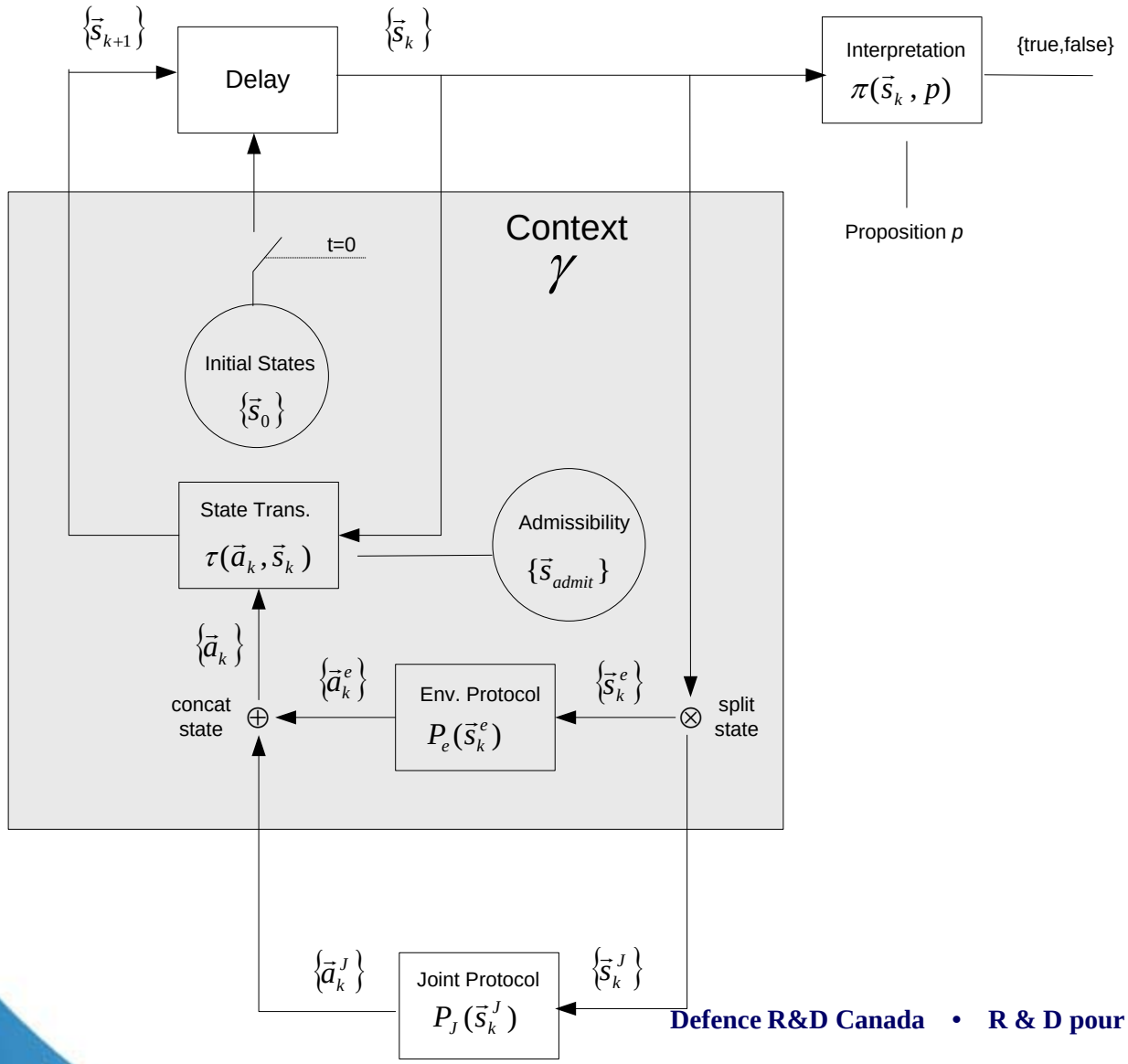


Situation

YES

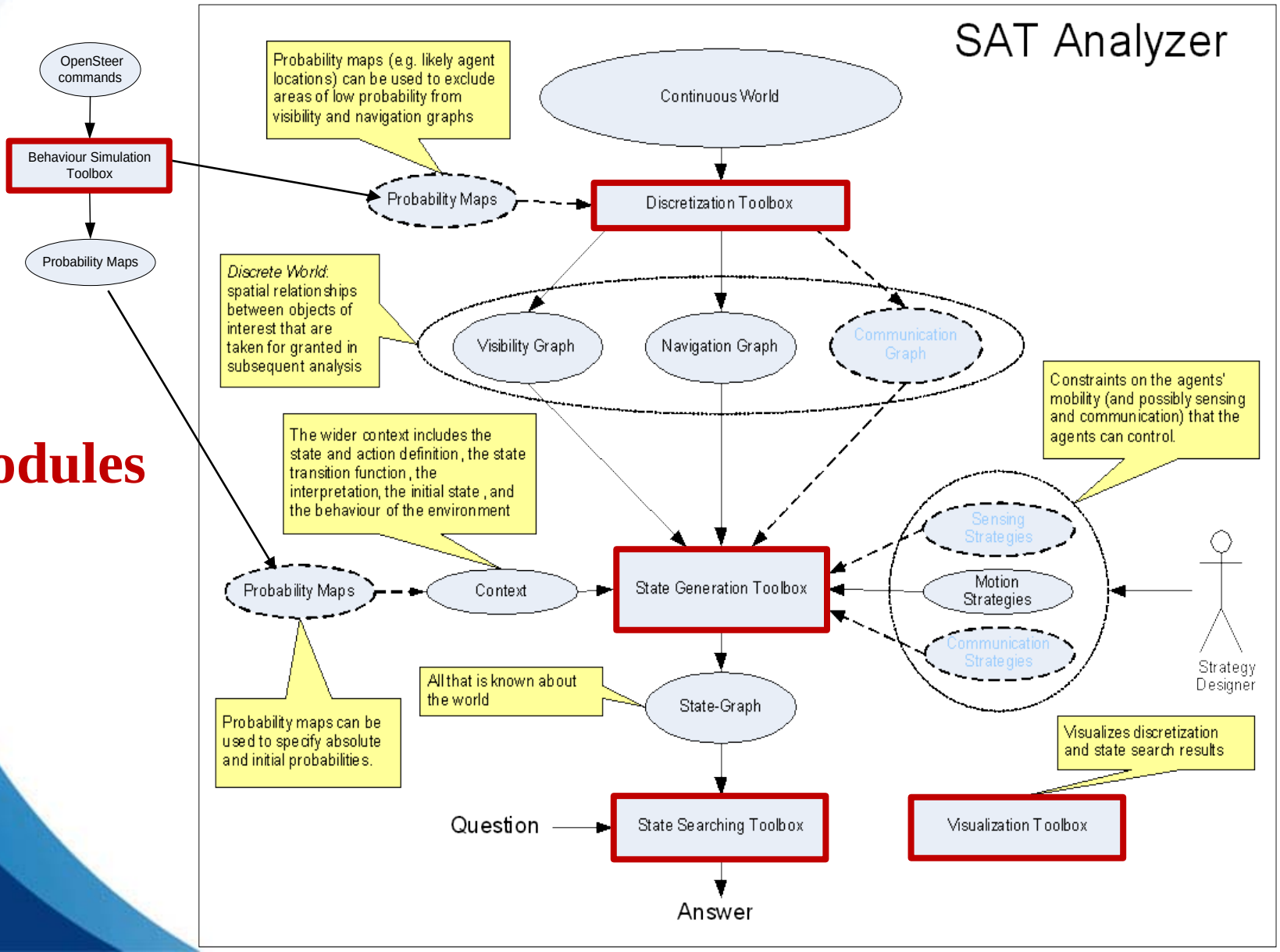
NO

Engineer view of Interpreted Systems



SAT – A Situation Analysis Toolbox

5 modules



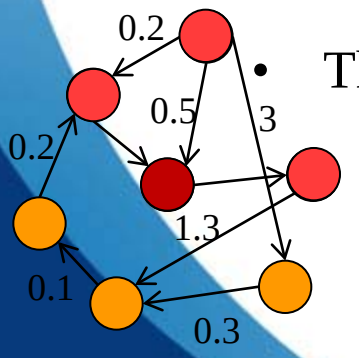
SAT – A Situation Analysis Toolbox

2 purposes:

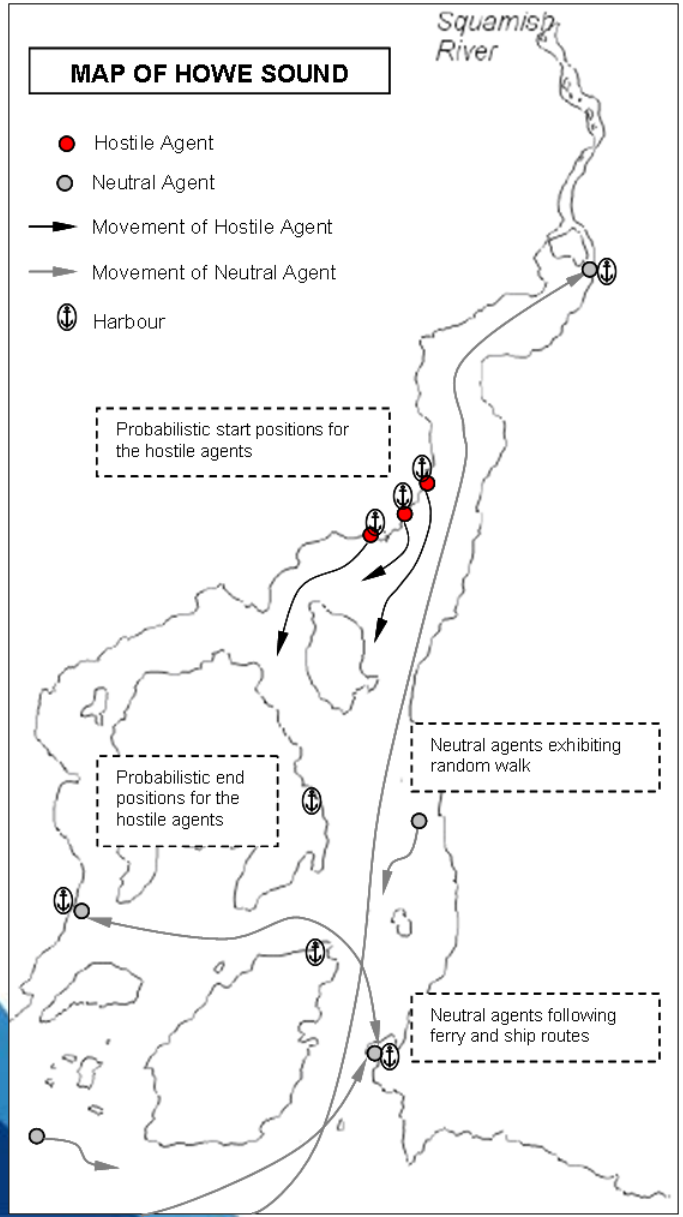
1. Situation **generation**
2. Situation **analysis**

Situations as pursuit-evasion (PE) games

- Pursuer and evader agents are constrained to move within a **graph** whose nodes are possible locations and whose edges denote paths between two locations.
- PE consists in a game between two teams having opposite goals, the players jointly seeking to maximize (resp. to minimize) a function of distance.
- Each agent has a **visibility** sensor of sensing range *r* meaning that the agent can see a node if it is within its range.
- Capture occurs when a pursuer and the evader are at the same position (node) at the same time.
- Basic action: Move from one node to an adjacent node.
- The evader's strategy P_e is unknown to the pursuers.



Counter-smuggling vignette



A smuggling operation has been reported in Howe Sound (north-west of Vancouver on Canada's West Coast).

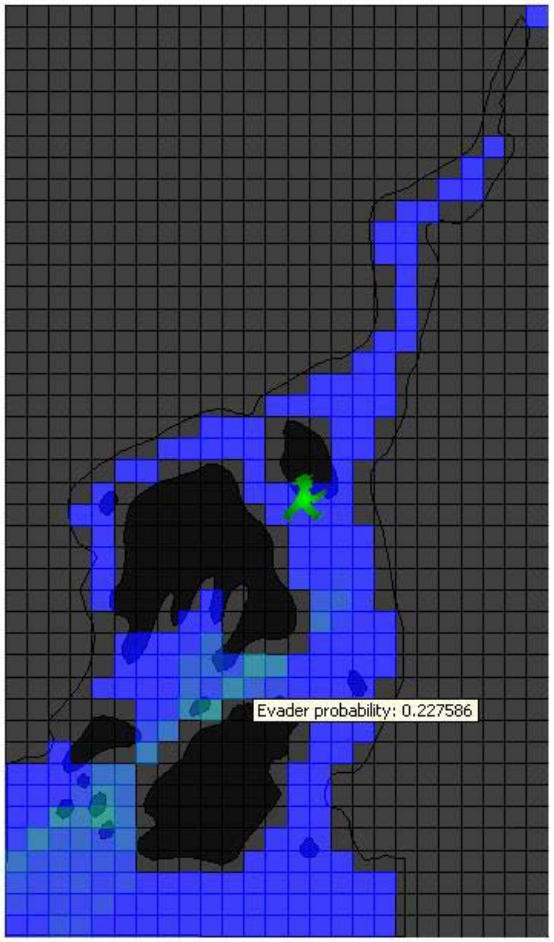
Can we guarantee that the smugglers will be detected ?

SAT – *Behaviour Simulation Toolbox*

Continuous world
(GIS map)

Partial knowledge
on evader's behaviour

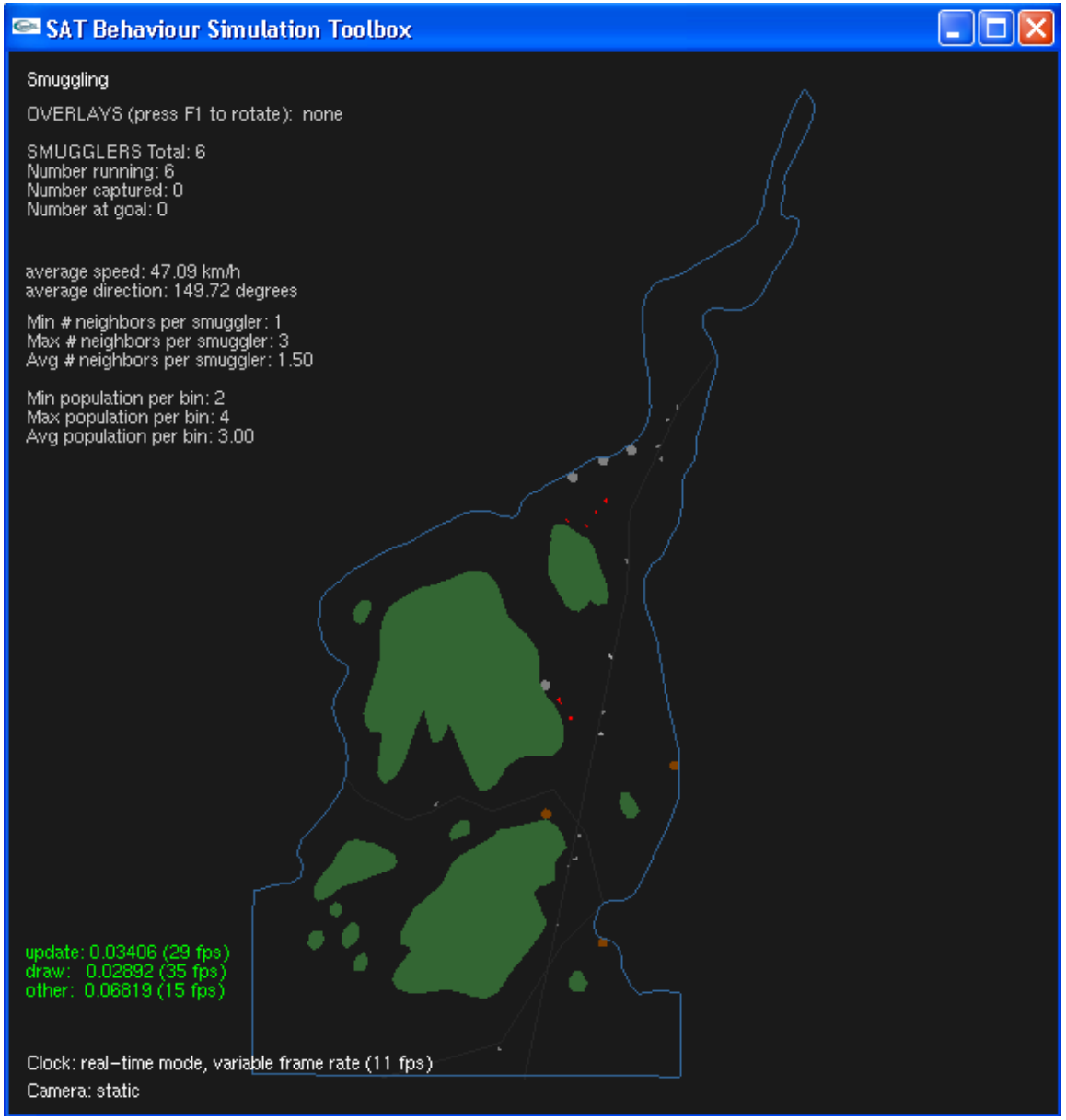
Behaviour simulator



Probability map
(agent containment)

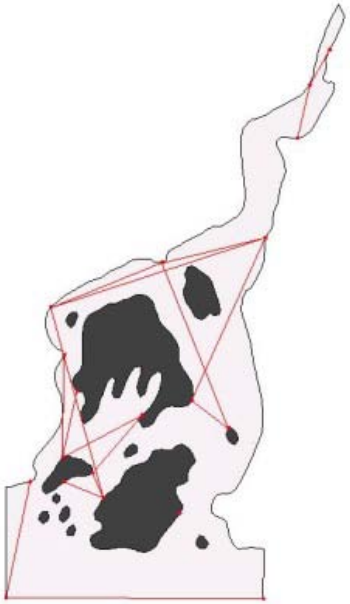
SAT – *Behaviour Simulation Toolbox*

Opensteer library



SAT – *Discretisation Toolbox*

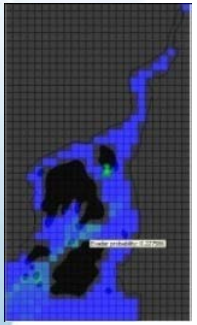
Discretiser



Visibility graph

Navigation graph

SAT - *The State Generation Toolbox*

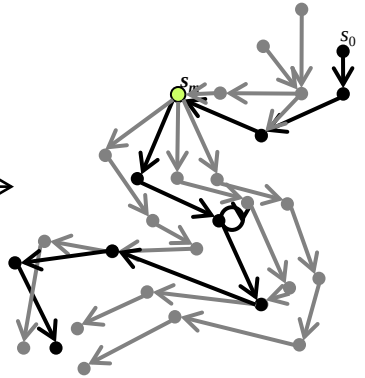


```
case of
   if PosP=PosE do Λ
   if PosE∈ Voisinage(PosP) do Move-to(PosE)
   else Move-to(n'importe quel nœud adjacent)
   Observe
end case
```

Context

γ

State generator



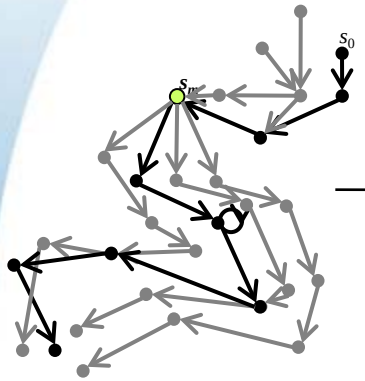
Situation
(state transition system)

```
case of
   if PosP=PosE do Λ
   if PosE∈ Voisinage(PosP) do Move-to(PosE)
   else Move-to(n'importe quel nœud adjacent)
   Observe
end case
```

P

Joint strategy

SAT – *State Searching Toolbox*



Situation

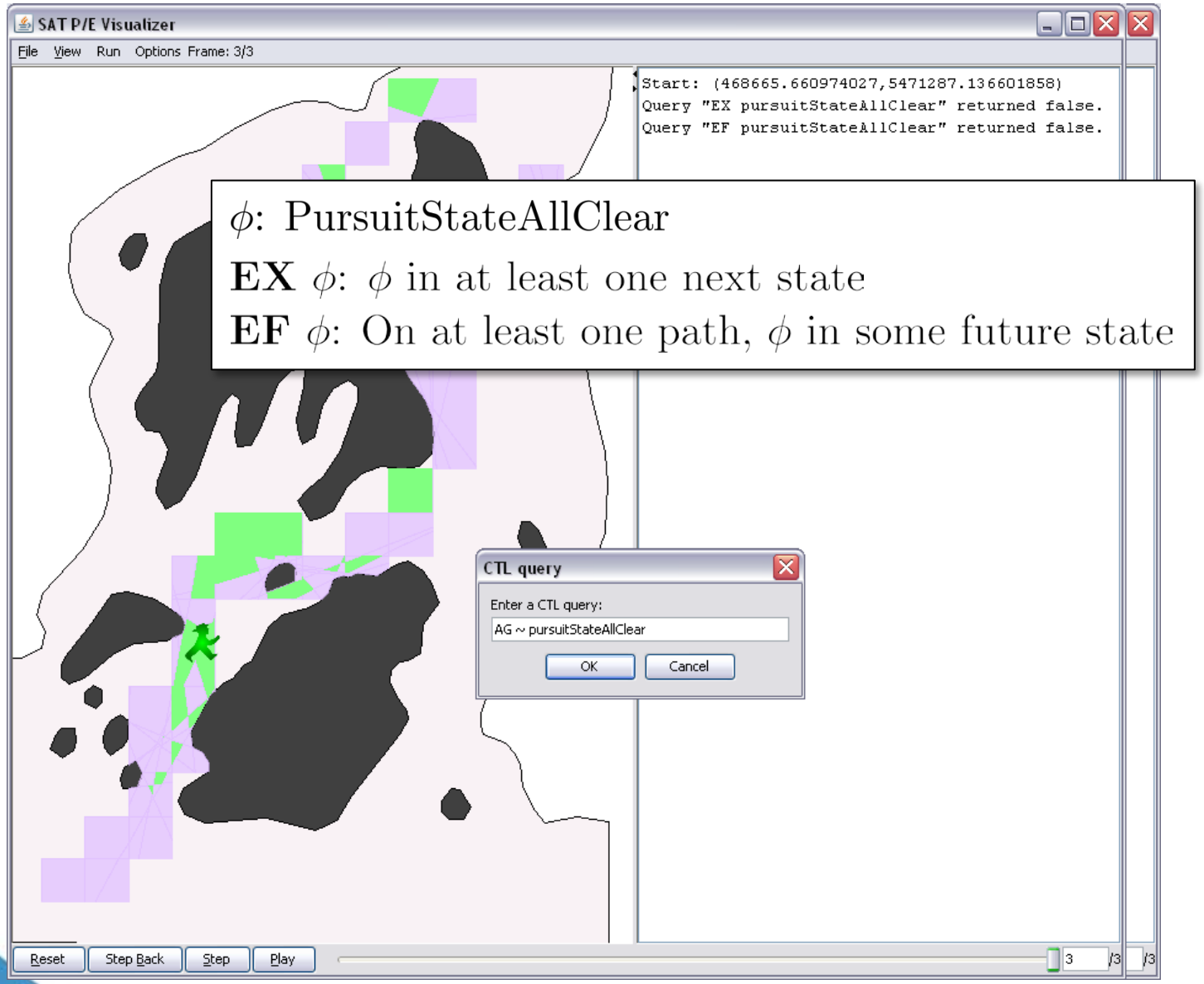
State searcher

Answer

CTL queries

AX ϕ	ϕ in all next states.
EX ϕ	ϕ in at least one next state.
A $[\phi$ **U** $\psi]$	on all paths, ϕ until ψ.
E $[\phi$ **U** $\psi]$	on at least one path, ϕ until ψ.
AF ϕ	On all paths, in some future state, ϕ.
EF ϕ	On at least one path, in some future state, ϕ.
AG ϕ	On all paths, in all future states, ϕ.
EG ϕ	On at least one path, in all future states, ϕ.

SAT – *Vizualisation Toolbox*



Conclusions

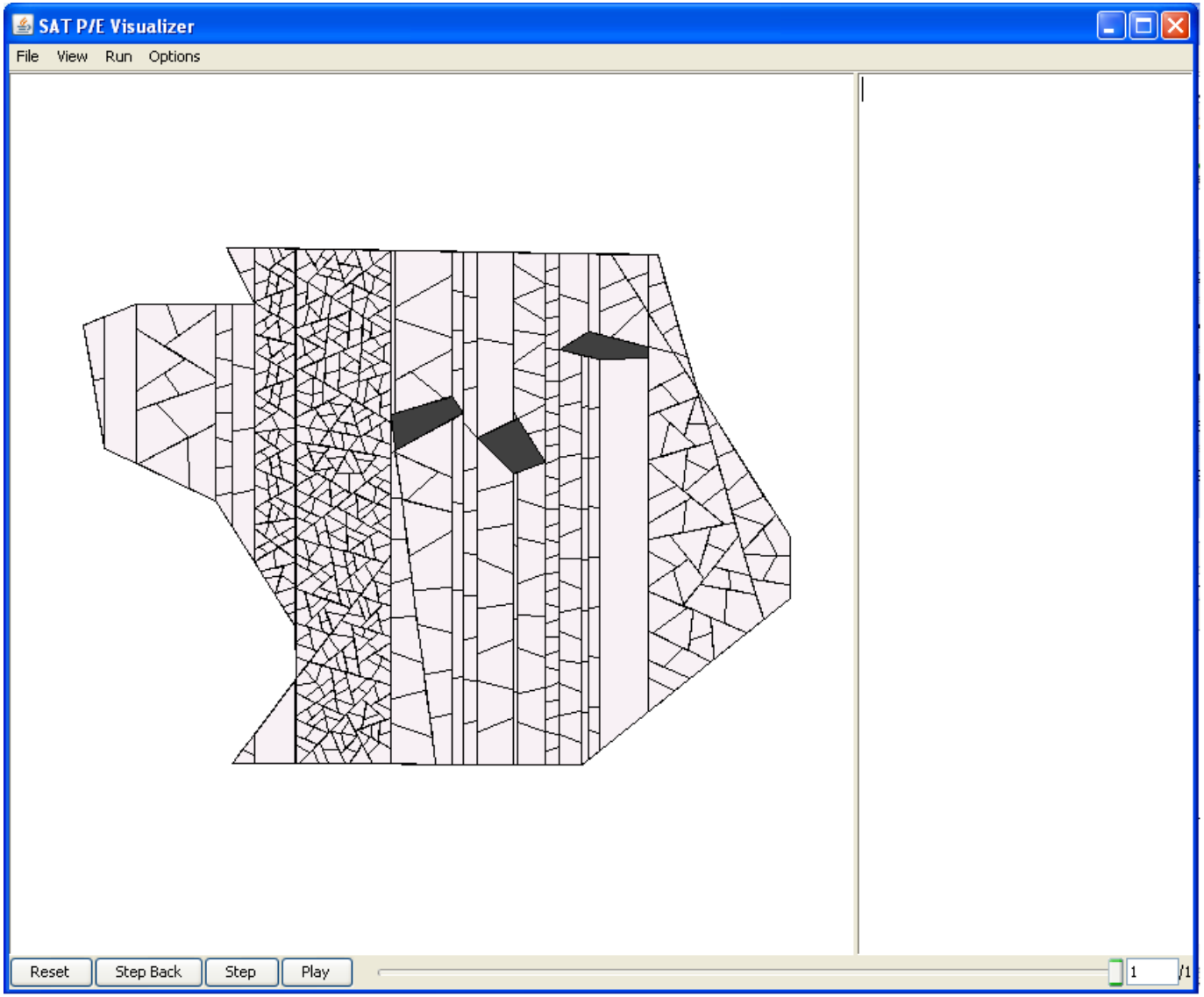
- The Situation Analysis Toolbox (SAT) implements formal notions of situation analysis based on epistemic state transition systems.
- The SAT **generates** situations based on
  1. an abstract version of the environment (visibility graph) enriched with probability maps of presence of agents, built through modeling emerging behaviour.
  2. the execution of a joint strategy derived from pursuit-evasion game theory.
- The SAT **analyses** the situation through logical queries.

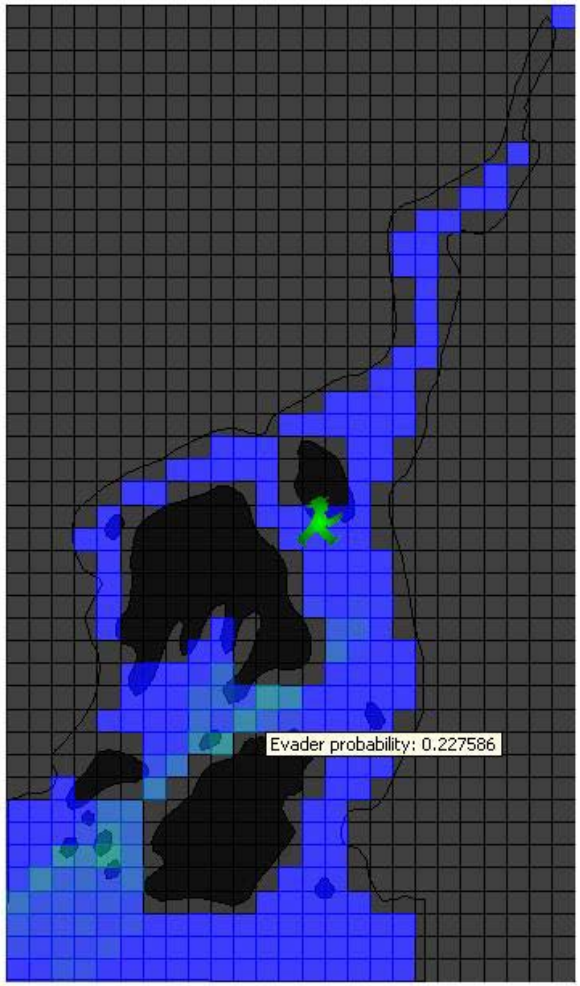
<u>Further works</u>:

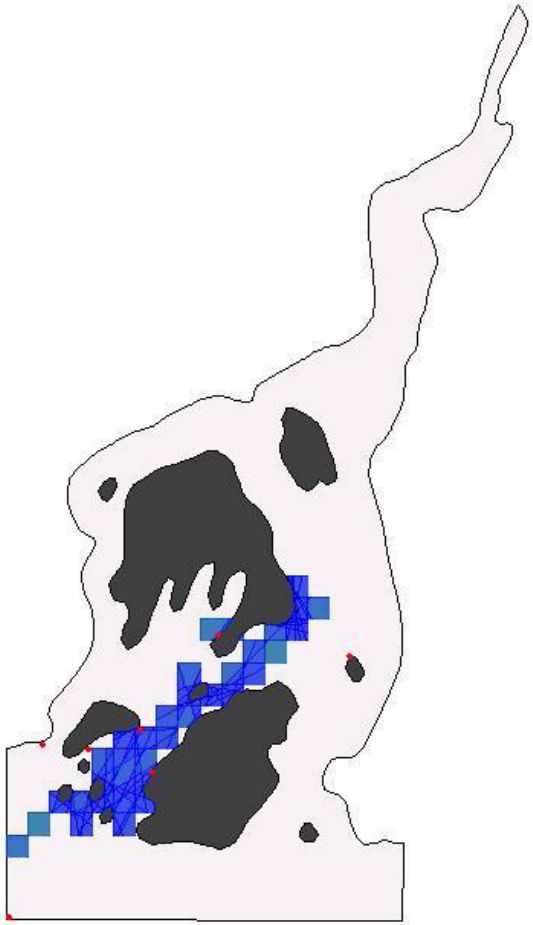
- Epistemic and probabilistic queries
- Customise to account for other applications


DEFENCE
R&D
DÉFENSE

SAT - *Vizualizer*







Probability map