Agent-based Simulation of the Maritime Domain

Ondřej Vaněk[1]

[1]Dept. of Cybernetics,FEE Czech Technical University, Technická 2, 166 27 Praha, Czech Republic

vanek@agents.felk.cvut.cz

Abstract. *In this paper, a multi-agent based simulation platform is introduced that focuses both on legitimate and illegitimate aspects of maritime traffic, mainly on intercontinental transport through piracy afflicted areas. Presented extensible architecture comprises of several modules controlling the simulation and the life-cycle of the agents, analyzing the simulation output and visualizing the entire simulated domain. The simulation control module is initialized by various configuration scenarios to simulate different real-world situations, such as a pirate ambush, coordinated transit through a transport corridor or coastal fishing and local traffic. The environmental model provides a rich set of inputs for agents that use both the geo-spatial data as well as vessel operational characteristics for their reasoning. The agents behavior model based on finite state machines together with planning algorithms allows complex expression of agent behavior, so the resulting simulation output can serve as a substitution of the real world data from the maritime domain.*

Keywords

Multi-Agent System, Simulation, Maritime Traffic

1. Introduction

Maritime domain is a complex environment consisting of many independent, however often intensively interacting entities that have their own goals and intentions. The aim of this research is to simulate not only the legitimate maritime traffic, such as an intercontinental transportation, coastal fishing or recreational traffic, but also the illegitimate aspects, such as illegal fishing, waste dumping and maritime piracy.

In this paper, an agent-based simulation of the maritime traffic is presented, more specifically the design of an architecture of the simulation platform is described (Section 2) and the simulation configuration and control flow explained (Section 4). A model of the maritime environment (Section 5) with several types of independent agents strive to achieve their goals (Section 6) is visualized using Google Earth[1] platform (Section 7).

The simulation is used to provide a noise-free source of maritime data with desired spatio-temporal resolution to

---
[1]http://earth.google.com

algorithms analyzing the situation and it serves for development and prototyping of agent-based techniques for understanding, detecting, anticipating and eventually preventing piracy and possibly other categories of maritime crime. For example, because of the recent surge in maritime piracy, insurance rates have increased more than 10-fold for vessels transiting known pirate waters over the past years and the overall costs of piracy in the Pacific and Indian ocean alone was estimated at US\$ 15 billion in 2006 and continues to rise [7] so it is convenient to develop a set of techniques and algorithms that are able reduce this threat. Description of these algorithms is out of scope of this paper, however it can be found in [5].

Although agent-based techniques have been successfully applied in other traffic and transportation domains and problems (see e.g. [4], [8]), this is – to our best knowledge – the first integrated attempt at employing agent-based concepts and techniques in the domain of maritime transport security.

2. Architecture Overview

The main objective in designing the simulation platform was modularity and extensibility; this objective has been met by employing a loosely coupled architecture with clearly defined data and control flows. A diagram depicting main modules and data flows is depicted on Figure **1**. As it can be seen, the simulation platform consists of several modules that can be arranged into the following groups:

- **Simulation Control** – modules contained provide the control of the initialization and execution of the simulation and all related processes (see Section 4).

- **Data Sources** – the modules contained provide the platform with off-line data required for initialization and run of the simulation. For description of data sources needed, see Section 3.

- **Simulation** – contains modules responsible for the representation and operation of the simulation model, both the simulated vessels and the simulated environment in which they operate.

- **Analysis** – contains modules performing analysis on the data coming from the simulation and a module emulating the imperfect aspects of the real world (noisiness
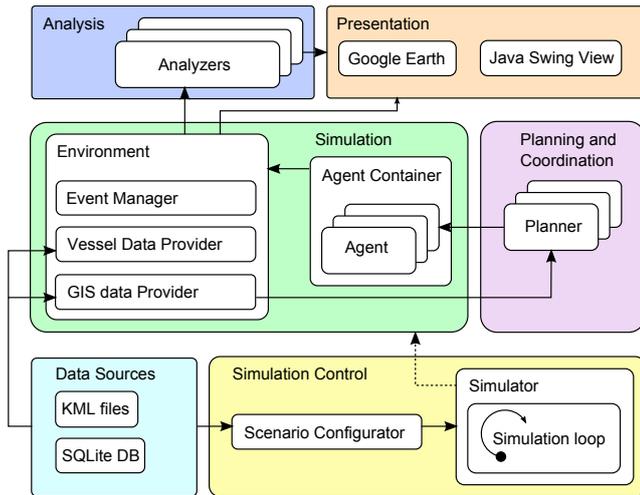
Fig. 1. Main modules and data flows. The lines represent main data flows, the dashed line represents the temporal synchronization controlled by the simulation loop.

and incompleteness of data). A detailed description of algorithms and modules in this category is out of scope of this paper.

- **Planning and Coordination** – contains modules responsible for more complex coordination and planning beyond the basic vessel behavior implemented as part of the simulation. Two different planners were so far implemented. A geo-spatial planner plans the vessel route with respect to geographical obstacles (coast, shallow water, etc.) and a planner using results of a game-theoretic formulation of the relation between the transport vessel and the agent (the formulation and results are briefly described in described in Section 6.3).

- **Presentation** – contains modules responsible for presenting the output of the simulation and the analytical modules using Google Earth KML[2] format.

3. Data Sources

As an essential feature, the platform incorporates several categories of real-world data and allows for their integrated analysis, specifically **general geographical data** that comprise general information about the geography of the environment, in particular shore lines, ports and shallow waters. These data are supplied directly by Google (Earth); they are used primarily for general vessel navigation and for providing background geographical context in the user frontend. **Operational geographical data** comprise geographical information specific to the operation of simulated vessel types, in particular the location of main piracy hubs [1], piracy zones, fishing zones and transit corridors. The oper-

ational geographical data govern the operation of individual vessel categories (see Section 6).

Vessel attributes describe vessel operational attributes such as vessel type, length, tonnage, max speed etc. These data are extracted from vessel tracking servers, e.g. AISLive [2] and are used to provide realistic parameters for simulated vessels. **Activity data** comprise higher-level information about maritime activity. These data are typically provided by organizations observing the situation in relevant regions. The Maritime Security Centre, Horn of Africa (MSCHOA) [3] provides up-to-date information on piracy incidents and alerts in the Gulf of Aden and off the Somali coast, including guidelines on how to proceed when traversing these areas. These data are used for pirate behavior modeling as well as for the route planning of transport vessels.

4. Simulation Control

The simulation is executed from a script with predefined configuration. The loading of the data and parameters and the creation of needed data structures is carried out in the Scenario Configurator module. The simulation time flow is synchronous and discrete, i.e. the simulation time is measured in steps, in each step a sequence of synchronized actions is executed.

4.1. Scenario Configurator

The Scenario Configurator handles the initialization of all modules and all agents present in the simulation.

The configuration of the simulation (e.g. the number of vessels, the files containing vessel and GIS data etc.) as well as the parameters of each of the modules is specified in a single GROOVY[3] script file[4]. The main advantage of this approach is that the script can be both embedded into a pre-compiled package and kept separate (in the source code form) to be able to run different scenarios or to modify the parameters of the simulation without recompilation.

The modularity of our architecture enables us to easily initialize only a small subset of Analyzers or to initialize a simulation without the support for KML-based visualization when running non-interactive batch experiments and logging the results for later post-processing.

4.2. Time flow

The main parameters governing the flow of the simulation are the *simulation step size* and *simulation step delay*. The *simulation step size* parameter specifies how much time one step takes, measured in simulation time (i.e. not related

---

[2]Keyhole Markup Language (KML) is an XML-based language schema for expressing geographic annotation and visualization

[3]http://groovy.codehaus.org/

[4]The nature of scenario description is more algorithmic than declarative; it is therefore more convenient to describe the scenario configuration using a script than using a markup language (such as XML).

to the real-world wall clock time[5]). The *simulation step delay* parameter specifies how much time a simulation thread sleeps after each step (to leave some computing time to other components, in particular the presentation and analysis component. It is measured in wall clock time. The *simulation duration* parameter specifies the total simulation time of a particular scenario.

The Simulator module holds information about the current simulation step and the total simulation time of the simulation run. This time is measured in simulation-time seconds (i.e. the maximal granularity of the simulation time is one second). Relating of the simulation time to date-anchored time is provided by the Environment module where the Environment module is initialized to a specific calendar time (i.e. $24th$ of December, 2009). Each time step, the Environment module updates the simulation time so the other modules can derive additional context information about relating to the given simulation date.

The amount of time in each time step (given by the *simulation step size* parameter) affects the simulation granularity and the temporal resolution of the output data obtained from the simulation. If the *simulation step size* is 1 s, the granularity is the finest and we can focus on vessel behavior in detail (e.g. vessel interaction in ports etc.). If the *simulation step size* is set to 1 hour, the simulation runs fast and we can roughly estimate the future positions of the vessels and possible events that will happen in the future.

4.3. Simulation Cycle

In each simulation step, the following sequence of actions is performed:

1. Environment module updates its simulation time.

2. Agent Container is notified and it sequentially sends the information about the new step to each agent. Each agent spends the amount of time on performing a part of its plan (see Section 6 for details). Generated events are sent to the Agent Container that distributes the events to all relevant listeners and to the Environment that records the event.

3. If – as a result of actions of the agents – new data become available, these are sent to the subscribed Analyzers for processing and analysis.

The synchronous nature of the simulation control together with seed-based randomization guarantees determinism and thus repeatability of the simulation. Deadlocks cannot occur as access to resources is sequential; this also enables the use of unsynchronized data structures which are faster to work with.

---

[5]*Wall clock time* is the human perception of the passage of time from the start to the completion of a task, i.e. the elapsed real-world time as determined by a chronometer such as a wristwatch or wall clock.

5. Environment Model

The state of the environment is maintained by two data modules – the GIS Data Provider and Vessel Data Provider and in a variable storing the current simulation time/date. The Environment module is a central access point for the other modules to the environmental data including data about the vessels.

The temporal information is represented in standard time and date units. The main units of spatial information are nautical miles; speed is measured in nautical miles per hour or per second.

GIS Data Model

The GIS Data Provider stores the geographical data and provides access to this information. The following data are represented and available for analysis and display:

- **Somali harbors** – a list of Somali harbors with name, location, description and approximate capacity for pirate vessels. These data are used for pirate vessels initialization and placement. The original file is a KML file.

- **Fishing zones** – a list of possible fishing zones around the Somali coast. The fishing area is represented by a polygon. These data are used for initialization of fishing vessels around the Somali coast. Currently, artificial zones are used; this can be later replaced by a list of real fishing areas if available.

- **Piracy zones** – a list of piracy zones around Somali coast. Each pirate vessel chooses a zone in which it is active. The zone selected is the closest one to the pirate's base harbor. Currently, artificial zones are used; this can be later replaced by a list of real piracy zones if available.

- **IRTC corridor** – data about the *International Recommended Transport Corridor*[6]. These data are used by the transport vessels to plan their trajectory.

Vessel Data Model

The Vessel Data Provider provides data about the ships in the simulation. Each record is about one vessel and is identified with a unique identifier, Vessel ID (this ID is equal to the ship's call sign when real-world ship parameters are used). Vessel-related data can be categorized into two groups:

- **Static vessel information** – static information stays the same throughout the simulation and cannot be modified. This information can be viewed as a database table

---

[6]as defined in `http://asianyachting.com/news/PirateCorridor.htm`

Vessel type	Parameters
Long-Range Transport Vessel	Start destination, Goal destination
Short-Range Transport Vessel	Start destination, Goal destination
Fishing Vessel	Home port, Fishing Zone
Pirate Vessel	Home port, Area of control, (targeted ship)

Tab. 1. Main parameters of different types of vessel agents.

where rows represents individual vessels and columns the attributes of each vessel. The data are stored in an SQL database for an easy access.

- **Vessel trace information** – a dynamic part of vessel information, storing the actual position of the ship as well as the previous positions in the form of time-annotated location records.

6. Agent vessel model

The agents reside in the Agent Container that distributes events gathered from the agents or from the Environment. Every agent controls one or more vessels. The plans for each vessel are either created prior to the simulation (e.g. for transport vessels) or, typically, generated dynamically during the simulation run (e.g. for pirate vessels).

6.1. Vessel Types

The platform can simulate the simultaneous activity of a large number (thousands) of the following categories of vessels: **Long-range transport vessels** are large- to very large-size vessels transporting cargo over long distances (typically intercontinental); these are the vessels that are most often targeted by pirate. **Short-range transport vessels** are small- to medium-size vessels which transport passengers and/or cargo close to the shore or across the Gulf of Aden. These ships are local and are not attacked by a pirate. **Fishing vessels** are small- to medium-size vessels which perform fishing within designated fishing zones; fishing vessels launch from their home harbors and return back after the fishing is completed. These vessels can be potentially attacked by a pirate, however currently only local fishing vessels that cannot not attacked, are simulated. **Pirate vessels** are small to medium-size vessels operating within designated piracy zones and seeking to attack a long-range transport vessel. The pirate control module supports several strategies some of which can employ multiple vessels.

Table 1 summarizes the main parameters of each type of vessel agents. Note that in general, a vessel agent can control more than one vessel (e.g. a mothership pirate vessel agent controlling several small boats and a hijacked transport ship).
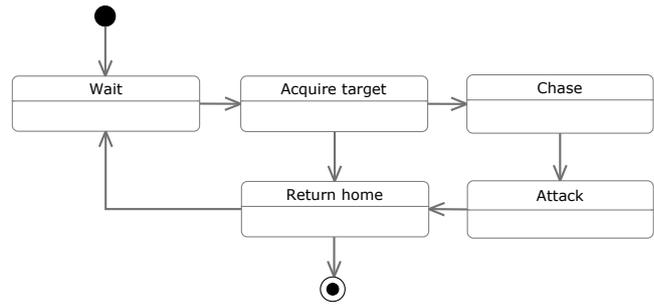

Fig. 2. Finite state machine of the pirate vessel agent.

The behavioral models for individual categories of vessels have been manually synthesized based on the information about real strategies obtained from several sources including IMB Piracy Reporting Centre[7] and Maritime Terrorism Research Center[8].

6.2. Executable Behavior Representation

Vessel agent behavior is implemented using finite state machines (FSM). Agent FSMs consist of states that represent agent's principal mental states. Transitions between the states are defined by unconditional or by conditional transitions conditioned by external events. Implementation-wise, the simulator allocates a time slice to the agent and the agent delegates the quantum to the FSM. The current state may either use the whole time slice and stay in the current state, or it can utilize only part of the time slice and delegate the rest of the time to a following state or states. An example of a pirate FSM is depicted on Figure 2.

6.3. Path Planning

A modular route planning architecture has been developed allowing to combine general shortest-route point-to-point planners with specialized planners for specific areas. More detailed description of the operation of both planners can be found in [5] and [6].

General Shortest-Route Navigation

The basic route planner finds a shortest route between two locations on Earth's surface considering vessel operational characteristics and environmental constraints, including minimum allowed distance to shore, which can differ between regions. The planner is based on the A* algorithm adapted for spherical environment and polygonal obstacles.

---

[7] http://www.icc-ccs.org/index.php?option=com_content&view=article&id=30
[8] http://www.maritimeterrorism.com/

Gulf of Aden Transit Planner

Factors other than route length need to be considered when planning the route through the Gulf of Aden. Two specialized planners have been therefore developed for this purpose. The simple corridor planner navigates the ship through the International Recommended Transport Corridor and mimics the current practice in the area. As a possible alternative, several alternative route planners through this area were implemented. A risk-minimizing route planner uses a pre-generated risk map to avoid high risk areas. A game-theoretic planner solves a two player zero-sum game between the transport ship and the attacker to find optimal path through the gulf. Detailed description can be found in [5].

7. Visualization

The user frontend provides the geo-based visualization of the various outputs provided by the platform, using Google Earth, a KML capable viewer.

Google Earth-based frontend allows to interactively visualize the various output of the testbed, both static data described in Section 3 and dynamically generated output of the advanced analysis and planning modules. A screenshot of the front-end is given in Figure **3**.

In addition to ergonomic navigation and 3d camera control, the main advantage of the front-end is the ability to present structured data on varying level of detail. The layer-based interface allows to select different layers of information and compose an information picture with the aspects and the level of detail fit for the specific user's need. To leverage the layer-based concept, the testbed output is organized into multiple information layers. The simulation itself provides a number of layers; each of the analysis and planning tools then also adds its own layer.

The integration with Google Earth is provided via dynamically constituted KML files served by an HTTP server running inside the platform. The KML files are read into the Google Earth application using its HTTP data link feature and automatically refreshed. This way, dynamic data can be displayed (such as a moving vessel), though for performance reasons, the refresh rate is limited to about once a second. Because of this and other limitations (e.g. in interaction with the user), it is possible to migrate the front-end to the Java-based NASA WorldWind platform[9] in the future.
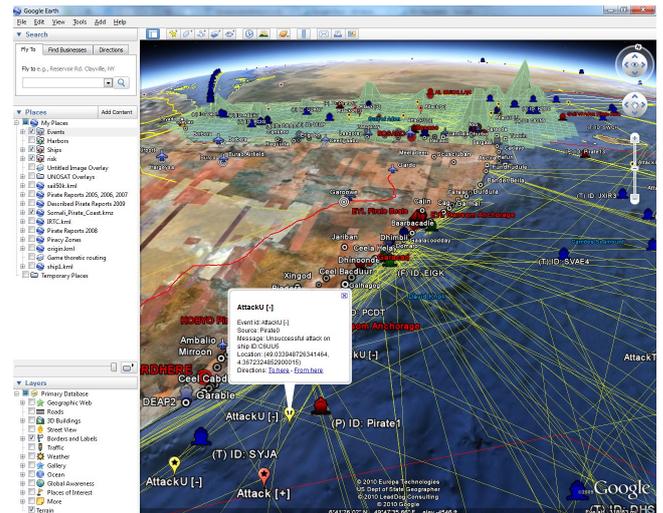
Acknowledgements

Fig. 3. Google Earth-based front-end showing vessels, their past trajectories and an output from dynamic incident risk analyzer over the Gulf of Aden.

References

[1] Somalia pirate attacks. BBS Google Earth Forum, http://bbs.keyhole.com/ubb/ubbthreads.php?ubb=showflat&Number=1242871%&site_id=1, April 2008.

[2] AISLive. http://www.aislive.com/, 2009.

[3] The maritime security centre, horn of africa (mschoa). http://www.mschoa.eu, 2009.

[4] GLASHENKO, A., IVASCHENKO A., RZEVSKI G., SKOBELEV P. Multiagent real time scheduling system for taxi companies. *In Proc. of 8th Int. Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS 2009)*, Budapest, Hungary, 2009.

[5] JAKOB M., VANĚK O., URBAN Š., BENDA P., PĚCHOUČEK M. Adversarial modeling and reasoning in the maritime domain, *Year 1 report*. Technical report, Agent Technology Center, Department of Cybernetics, FEE, CTU Prague, 2009.

[6] JAKOB M., VANĚK O., URBAN Š., BENDA P., PĚCHOUČEK M. AgentC:Agent-based testbed for adversarial modeling and reasoning in the maritime domain. *Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010) - demotrack*, May, 2010.

[7] NINCIC D. Maritime piracy in Africa: The humanitarian dimension. *African Security Review*, vol. 18(3): p. 216, 2009.

[8] PĚCHOUČEK M., ŠIŠLÁK D. Agent-based approach to free-flight planning,control, and simulation. *IEEE Intelligent Systems*,vol. 24(1): p. 1417, Jan./Feb. 2009.

About Authors. . .

Ondřej VANĚK graduated from Technical Cybernetics in 2008 at FEE, CTU and is a PhD student at the Agent Technology Center at the Department of Cybernetics, FEE, CTU. His main research is focused on machine learning in multi-agent systems, cooperative and non-cooperative game theory and coordination and cooperation in multi-agent systems.

---

[9]http://worldwind.arc.nasa.gov/