Auslander, B., Molineaux, M., Aha, D.W., Munro, A., & Pizzini, Q. (2009). *Towards research on goal reasoning with the TAO Sandbox* (Technical Report AIC-09-155). Washington, DC: Naval Research Laboratory, Navy Center for Applied Research in Artificial Intelligence.

Towards Research on Goal Reasoning with the TAO Sandbox

Bryan Auslander[1], Matthew Molineaux[2], David W. Aha[1], Allen Munro[3] & Quentin Pizzini[3]

[1]Navy Center for Applied Research in Artificial Intelligence;
Naval Research Laboratory (Code 5514); Washington, DC 20375
[2]Knexus Research Corporation; Springfield, VA 22153
[3]Rossier School of Education; University of Southern California; Redondo Beach, CA 90277
first.surname@nrl.navy.mil | matthew.molineaux@knexusresearch.com | *surname*@usc.edu

2009 September 15

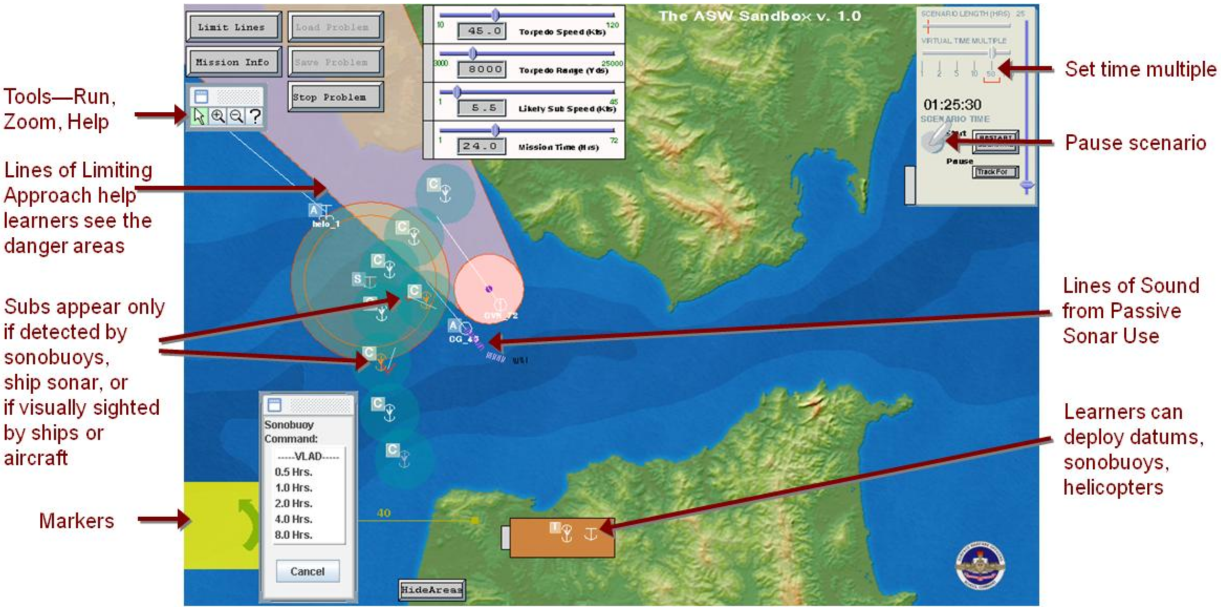
Abstract. We describe our progress on instrumenting a Navy software simulator for use in the context of intelligent agent research. The Tactical Action Officer (TAO) Sandbox, which is being developed at the University of Southern California, is used by officers to practice tactical decision making in the context of Navy surface fleet missions. NRL and Knexus Research Corporation have integrated this simulator with intelligent agents using the Lightweight Integration and Evaluation Testbed (LIET), thus permitting the agent to play the role of a trainee. This will permit us to use the TAO Sandbox in our artificial intelligence research, where we are currently focusing on algorithms for continuous planning that can dynamically reason about what goal should be pursued at any time during a mission. This paper briefly descibes our motivation for this integration, project status involving this simulator, and future goals.

1. Motivation

One of the core topics within the artificial intelligence (AI) research community is *planning*, which concerns the task of synthesizing a *plan* to accomplish a *goal* from an initial *state* using a set of *actions* (Russell & Norvig, 2003; Ghallab *et al*., 2004). In this context, a state space S represents the entire set of possible states, an initial state $s_i \in S$ is a point in this space, a goal $G \subseteq S$ is a set of one or more points in this space, and an action $a \in A$ such that $a: S \rightarrow S$ is a mapping among states (i.e., an action permits movements among states). Thus, a planner $p(G, s_i, A)$ identifies a set of actions $a \in A$ that can potentially be executed so as to traverse from s_i to some state in $g \in G$.

Planning is an old topic dating to the earliest days of AI, and hundreds (if not thousands) of papers have been devoted to it, ranging from theoretical to applied, automated to interactive, and involving a large variety of approaches and assumptions. Also, annual international conferences are now devoted to AI planning. Practical motivations for planning abound, from industrial robotics tasks to Navy needs (e.g., controlling automated vehicles, assisting with assessing Course-of-Action plans, and providing intelligent agents – neutrals, allies, or adversaries – in training simulations). However, although a few exceptions exist (e.g., Cox, 2007), comparatively little research has been devoted to developing methods that can dynamically/automatically generate and manage their own goals, which is a desired capability of intelligent agents for continuous planning tasks (e.g., such as those involving many Navy missions).

Instrumented simulators are valuable for AI research because they can provide a source for data and evaluation metrics when integrated with intelligent agents (Molineaux & Aha, 2005). We seek a simulator that can serve as an *environment* (i.e., for accepting actions as input, executing them, and providing state information as output) for conducting continuous planning research on dynamic goal generation and management. Furthermore, we prefer to use Navy-relevant simulators in our research. In the following sections, we describe such a simulator (the TAO Sandbox) and its intended use in this context.



Torpedo Speed (Kts) 45.0

Torpedo Range (Yds) 8000

Likely Sub Speed (Kts) 5.5

Mission Time (Hrs) 24.0

SCENARIO TIME 01:25:30

Figure 1: Snapshot of the TAO Sandbox user interface, which can be used by instructors to create simulated ASW scenarios involving the deployment and control of helicopters, sonobuoys, ships, and sonar systems. Trainee solutions provided are recorded for later review by instructors.

2. TAO Sandbox

2.1 Features of the TAO Sandbox

The TAO Sandbox (TAOS) (Munro & Pizzini, 2009) is an interactive Navy training simulator for Tactical Action Officers that is being developed by Allen Munro and Quentin Pizzini at USC's Center for Cognitive Technology. This system was designed with guidance from the Surface Warfare Officer's School (SWOS) in Newport, RI, and has been used (as a series of evolving implementations) in the SWOS Department Head Course since 2008. Trainees, working in teams, interact with the TAOS to solve conceptual problems in simulated anti-submarine scenarios (that were previously authored by instructors), during which they address decision-making situations (e.g., requiring ship coordination in a hostile environment) similar to ones that they may need to resolve later in operational situations. Trainee interactions can be recorded and replayed to assess the students' performance. The TAOS can (1) provide clear demonstrations of ASW issues and tactics to trainees, (2) supply more (and more varied) ASW problems to instructors, and (3) allow users to create new scenarios that may require new tactics. Figure 1 highlights aspects of the TAOS user interface and system in Problem mode (as distinguished from Instructor Mode, which is used to create and edit scenarios).

A TAOS scenario includes a map containing a human-operator controlled group of ships, planes, and helicopters, along with AI-controlled enemy submarines and miscellaneous other ships that may be hostile. The operator's task is to use their units to accomplish certain mission objectives such as:

- Locate and engage enemy submarines
- Identify all ships in an area to find a target ship
- Escort a Mission Essential Unit (MEU) to a destination (e.g., a naval base)

These scenarios can be simulated to take place in any part of the world by changing or adding a map of the desired area. USC is creating more vehicle actions and adding more types of units, which further increases the system's real-world applicability and allows for more in-depth experiments.

The Navy Center for Applied Research in AI (NCARAI) at the Naval Research Laboratory is currently working with this simulator as a test-bed for current and future research. NCARAI seeks Navy simulators, such as the TAOS, for conducting AI research. TAOS is an excellent simulator to work with, due to its real-world application, use of limited knowledge (e.g., submarine locations, ship identities), and software system stability.

From the perspective of AI researchers, it is useful to characterize novel environments, such as the TAO Sandbox, so as to understand what types of scenarios it can support. To do this, we use the six properties of task environments suggested by Russell and Norvig (2003) in their popular AI textbook:

1. *Fully vs. Partially Observable*: The TAOS is a partially observable environment, meaning that not all state information is available to every agent. For example, the friendly forces may not necessarily have location information on enemy submarines, nor even awareness of their proximity.
2. *Deterministic vs. Stochastic*: The TAOS is a deterministic simulator. For example, from a given state, the same outcomes for sensing and ship movement will result each time that same (exact) state is visited.
3. *Episodic vs. Sequential*: The TAOS is a sequential simulator. For example, a launched torpedo can have significant effects in future states.
4. *Static vs. Dynamic*: The TAOS is dynamic in that, if the simulator continues to execute while the agent is deliberating on its next action(s), then the state can change (e.g., movements of enemy submarines). However, we are currently stopping the environment during agent deliberation, thus treating the environment as static.
5. *Discrete vs. Continuous*: States in the TAOS include continuous dimensions, such as the location of objects in the environment. In addition, events (e.g., ship movements) take place continually over time.
6. *Single Agent vs. Multiagent*: The TAOS is a multiagent environment. For example, neutral and enemy ships and submarines can also be controlled by other agents.

In addition, TAOS can be characterized as having moderately complex physics, although it is not completely 3-dimensional (e.g., submarine depth is not represented, nor are helicopter altitudes). However, the simulator does attempt in most instances to simulate vehicle vectors and their changes. For instance vehicles in the simulator do not turn instantly, but rather incrementally over time, meaning that angular velocity must be accounted for in some way. For example, if a torpedo is fired from dead ahead of the ship, then it is unlikely for that ship to be able to outrun the torpedo since it cannot turn fast enough. In addition vehicles in the environment all have different maximum speeds, sensor capabilities, and torpedo ranges allowing for complex spatial reasoning about which unit is appropriate for which task.

TAOS implements a wide range of sensors for agents describing the game world. Before describing the simulator‘s sensors, we should mention that it lacks the ability to detect land. The TAOS was designed with human operators and instructors in mind who can intuitively process the background image and avoid land masses. For intelligent agents to do the same, one solution is to run experiments taking place solely in open ocean. In the future, it may be possible to perform a priori image analysis to detect land masses.

The majority of TAOS sensors concern vehicle positioning and status essential for an automatic agent to maintain the world state:

- Vehicle Locations
- Vehicle Velocity
- Vehicle Status (destroyed)

However, the sensors of primary interest are those that can detect submarines (i.e., mainly passive and active sonar, along with radar). Ships enabled with passive sonar can detect submarines within passive sonar range while travelling at speeds under 15 knots with the sonar enabled. This ship-speed constraint is used to simulate how faster speeds can decrease sonar performance due to the extra noise. When a submarine is detected with passive sonar a line of sound (LoS) is drawn originating from the ship that crosses through that submarine. In the simulator as in the real world using LoSs from multiple bearings makes it possible to determine a submarine's heading and, possibly, its location. The TAOS level of accuracy though, with LoSs going through the submarines, appears to be more precise than its real-world counterpart, allowing for easier submarine pinpointing.

Unlike passive sonar, active sonar will pinpoint the location of a submarine within range of a ship using it. However, in most scenarios active sonar has a significantly smaller range than passive sonar. Therefore, passive sonar is used first to obtain a general location of the submarine. Active sonar on ships also have a small cone area in the back of the ship that it cannot sense (helicopters do not have this problem). The active sonar seems to adhere well to what it represents, although the implementation for both types of sonar has made some simplifications. For instance, there are no underwater obstacles considered that the submarine could use to hide behind, and there is no chance for a false positive reading.

Radar does not have a huge use in the scenarios that we have been studying. Currently, its only function is to detect a submarine that has raised its periscope. Nonetheless, this information is invaluable to the *mission essential unit* because it may provide an early warning of a sub closing in for an attack, as submarines ***must*** first identify their target with the periscope before shooting a torpedo.

In our scenarios, we use a *single* agent to control *all* of the friendly units (e.g., ships, helicopters, planes, sonobuoys). There is a wide variety of TAOS actions for each unit type within this agent's control. These include:

- Changing bearing and speed
- Turning on and off different sonar and radars
- Launching helicopters
- Dropping torpedoes
- Dropping different types of sonobuoys
- Turning on and off sonobuoys

For the most part these actions are executed with only a little bit more control than one would expect from a real-time strategy game. For example, helicopter flight mechanics are (for the most part) automatically processed by the simulator; the user only specifies speed and bearing. Similarly, acceleration is not modeled; ships move either start at (and maintain) full speed or stop instantly. Another interesting issue is the ability to drop unlimited amounts sonobuoys anywhere on the map. However, this is most likely a temporary solution until, in future TAOS versions, planes or helicopters are needed to drop them.

While useful for the types of near-term research investigations we seek to conduct, the TAOS has some limitations for research that should be mentioned. For example, we have not yet succeeded in speeding up the execution of scenarios to permit hundreds, if not thousands, of scenario executions at high speed. Unless corrected, this will limit the scope of our empirical studies (on traditional CPUs). Also, the inability to detect land masses limits the scenarios available to the automatic agents. Future support will be needed to conduct experiments that may involve limited maneuvering room (e.g., inside of a straight).

The current simulator also limits all experiments to a task force that contains some combination of the five main ships, their helicopters, and a single plane. The number of submarines has also been limited to two. Finally, the submarines will attack *only* the mission essential unit. Thus, some scenarios cannot be represented, such as one where submarines attack merchant (neutral) ships in trade lanes.

2.2 Implementation of the TAO Sandbox

The TAO Sandbox is being implemented using iRides Author (Munro 2003; Munro, Surmon, and Pizzini, 2006), a development environment sponsored by the Office of Naval Research under a series of grants and contracts to the University of Southern California. This development environment provides separate facilities for developing simulations and scripted instructional vignettes in the context of simulations. Simulation specifications and instructional specifications are stored in separate files of different types, and each specification is interpreted by its own execution engine. The instructional engine makes use of services offered by the simulation engine. Such services include highlighting objects, registering an interest in expression value changes, setting values in the simulation, pausing and restarting the simulation, and so on. In order to have access to these facilities, the instructional component registers itself with the simulation engine as a *Simulation Listener*. This facility has been exploited by NCARAI to create a software interface for the system, as described in Section 3.

A lightweight version of iRides Author, called iRides, is used to deliver interactive simulations and scripted instructional vignettes to learners. Both iRides and iRides Author were developed as Java applications. iRides can also be used as a Java applet, although this is only possible for simulations that do not require file access as part of their behaviors. The TAO Sandbox does require access to files that store scenario data and session recordings of simulation actions, so it can be used only with the application versions of iRides.

3. Current Research

3.1 The TAOS Interface

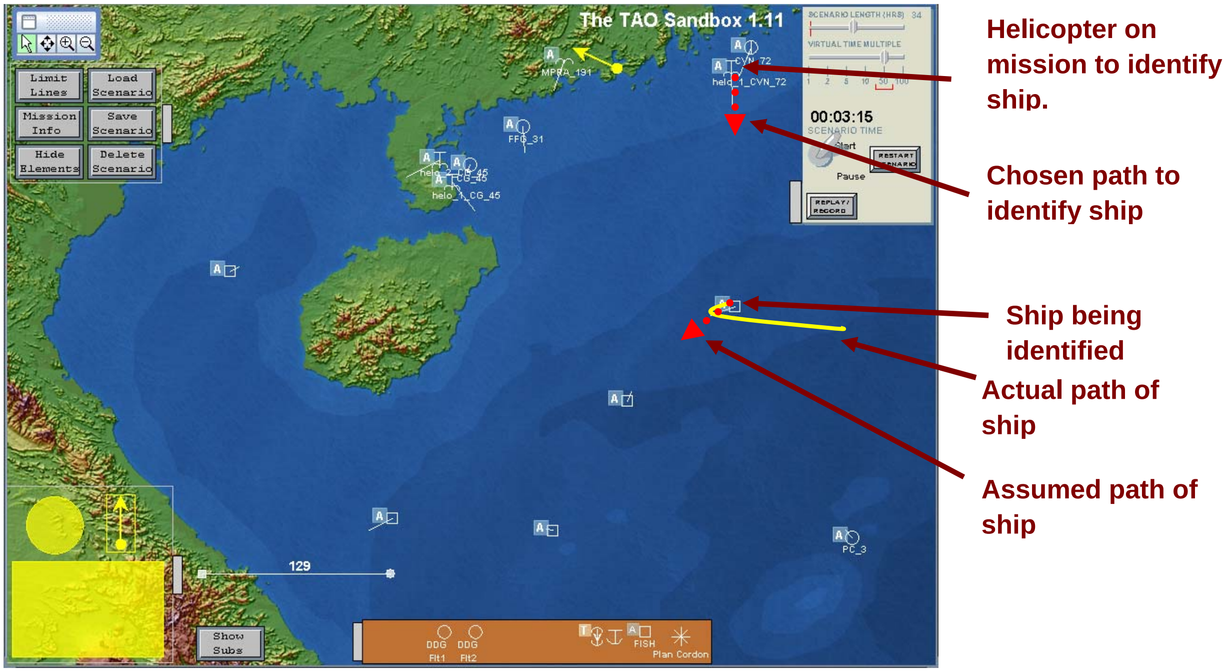
To date, most of NCARAI's work with the TAOS has concerned creating a software integration interface for the system. The simulator was originally designed with a human operator in mind with minimal support for automated AI. Even the computer controlled units in TAOS do not have much intelligence; they tend to travel on preset paths until they come within range of a human-controlled ship, where upon they may fire a torpedo. Given this, we created an interface for an intelligent software agent to control the operator's vehicles, and to obtain observations about the environment's state.

We designed this TAOS interface so that most necessary actions available in the system are now mapped to different methods for a controlling agent. This allows for incremental development with the TAOS and eases maintenance by being able to pinpoint computation errors. The current interface can perform all of the required actions in TAOS version 1.11. However, our interface is not entirely compatible with newer TAOS versions, which have added features (e.g., new units such as air bases, missile launchers, and surface to air missiles) and changed some functionality.

3.2 TAOS LIET Integration

We integrated the TAOS with LIET (Molineaux *et al.*, 2009), which is a free tool developed by Knexus that can be used to integrate simulation environments (such as the TAOS) with intelligent agents. LIET also provides facilities to perform experimentation and performance analysis. We used a modified version of the SHOP2 Hierarchical Task Network Planner (Nau *et al.*, 2003) and a custom-built TAO domain as the initial agent. Our modification to SHOP2 enabled *temporal* planning, which is essential for real-time environments like the one supported by TAOS simulator.

The TAOS-LIET integration has been successful. LIET can start up the TAOS, read state information from it, and convert this information into correct LISP syntax for our planning agent. LIET can also convert the plan created by this agent into actions that the TAOS simulator can execute.



Helicopter on mission to identify ship.

Chosen path to identify ship

Ship being identified

Actual path of ship

Assumed path of ship

Figure 2: Snapshot of a TAO Sandbox scenario executing a plan to identify unidentified ships on the map. As indicated by the yellow line, this scenario includes a surprise course change for the ship being identified that will cause an aerial unit to miss it during its attempt to identify the submarine.

3.3 Example Scenario for Dynamic Goal Reasoning and Management

TAOS offers a large playground for research on intelligent agents due primarily to the open-endedness of the goals in the system along with the ease in which one can create a custom scenario. This means that most reasonable scenarios are only limited by what objectives the researchers conceive. One simple scenario that requires dynamic goal reasoning and management is the task of identifying a ship that changes course to avoid being identified. In this scenario the operator needs to send a plane or a helicopter to make visual contact with an unidentified ship. In simulation terms this means the aerial unit needs to be within the visual range of the ship as defined by the current scenario. When planning for identifying a ship there is no way of knowing what the ship's course is going to be in the future; the only information available to the planner is the ship's current speed and bearing. Therefore, it is possible for a ship to make an erratic course change that would prevent the aerial vehicle from making visual contact along its flight path.

An example of this scenario is shown in Figure 2, where a helicopter was assigned to identify a ship that will soon (after the assignment) make a 180° turn, causing the helicopter to not fly into its visual range. When the plan ends or is updated the expectation (that the ship will be identified) will fail, and dynamic reasoning will be required to fix the situation. An explanation that the ship changed course should be generated along with a plan to fix the inconsistency.

Some other simple situations that require similar dynamic reasoning are:

- While searching for an enemy submarine one of the operator-controlled ships is destroyed by a torpedo.
- While identifying ships a hostile or target ship is spotted.
- A submarine sighting is reported.

4. Future Work

We integrated the TAOS with LIET to conduct research on dynamic goal reasoning and management. Through iterative system testing, we will increase the complexity of the HTN plans that our agent can process, and conduct a systematic evaluation of its components.

We also plan to use the TAOS in future research projects, including one focused on assessing the utility of relational data representations and another on studying the utility of different types of spatial-temporal representations for supporting models of computational analogy.

Acknowledgements

This work was sponsored as a DARPA/IPTO seedling project via MIPRs 09-Y213 and 09-Y214. Thanks to Program Manager Michael Cox for providing technical direction that motivated this work. The views, opinions, and/or findings contained in this article/presentation are those of the author/presenter and should not be interpreted as representing the official views or policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the Department of Defense.

Bibliography

Cox, M. T. (2007). Perpetual self-aware cognitive agents. *AI Magazine*, **28**(1), 32-45.

Ghallab, M., Nau, D.S., & Traverso, P. (2004). *Automated planning: Theory and practice*. San Mateo, CA: Morgan Kaufmann.

Molineaux, M., & Aha, D.W. (2005). TIELT: A testbed for gaming environments. *Proceedings of the Twentieth National Conference on Artificial Intelligence*. Pittsburgh, PA: AAAI Press.

Molineaux, M., Aha, D.W., & Sukthankar, G. (2009). Beating the defense: Using plan recognition to inform learning agents. *Proceedings of the Twenty-Second International FLAIRS Conference* (pp. 337-343). Sanibel Island, FL: AAAI Press.

Munro, A. (2003). Authoring simulation-centered learning environments with Rides and Vivids. In Murray, T., Blessing, S., and Ainsworth, S. (Eds.) *Authoring Tools for Advanced Technology Learning Environments*. Dordrect, The Netherlands: Kluwer Academic Publishers.

Munro, A., Surmon, D., & Pizzini, Q. (2006). Teaching procedural knowledge in distance learning environments. In H.F. O'Neil & R.S. Perez (Eds.), *Web-based learning: Theory, research, and practice* (pp. 255-278). Mahwah, NJ: Lawrence Erlbaum Associates.

Munro, A., & Pizzini, Q. (2009). The TAO Sandbox user guide. Unpublished technical report.

Nau, D., Au, T.-C., Ilghami, O, Kuter, U, Murdock, J.W., Wu, D., & Yaman, F. (2003). SHOP2: An HTN planning system. *Journal of Artificial Intelligence Research*, **20**, 379-404.

Russell, S., & Norvig, P. (2003). *Artificial intelligence: A modern approach* (2nd ed.). Upper Saddle River, NJ: Prentice Hall.