Adaptive Agents for Adaptive Tactical Training: The State of the Art and Emerging Requirements

Jared Freeman[1(✉)], Eric Watz[1], and Winston Bennett[2]

[1] Aptima, Inc., 12 Gill Street, Suite 1400, Woburn, MA 01801, USA
freeman@aptima.com
[2] Warfighter Readiness Research Division, 711 HPW/RHA, USAF Dayton, OH, USA

Abstract. Military tacticians require practice to learn their craft. Practice requires adaptive opponents capable of responding to trainee actions in ways that are realistic and instructionally productive. Current agents are generally too brittle, too scripted and unresponsive to support adaptive training in this way. What is required to develop adaptive agents are (1) real-time feeds of simulator data that are sufficiently rich and realistic to support agent development and execution; (2) agent architectures capable of generating realistic and instructive behaviors from these data; and (3) a testbed that can deliver data and performance measures in sufficient volume to enable modelers to accelerate agent development by applying emerging analytics and machine learning. The 711th Human Performance Wing/RHA has invested in precisely these capabilities over four years, engaging eight of the leading developers of intelligent agents. In this paper, we describe these capabilities, and, importantly, the data requirements these solutions impose on simulators and operational systems that can employ these technologies in the future.

Keywords: Adaptive training · Cognitive models · Tactical training

1 Introduction

U.S. Air Force pilots battle smart, agile enemy flyers in the wild. But in training simulations, the automated enemies – Computer Generated Forces (CGF) – are often predictable in flight and unresponsive to maneuvers by pilot trainees. Pilots can rehearse textbook tactics against textbook adversaries under these circumstances. That has high value early in pilot training. However, when pilots apply subtle, novel, or erroneous tactics, CGF typically respond unrealistically or do not respond at all. Human simulator operators often take over from CGFs when smart, agile behavior is needed. This work-around raises the staffing cost of simulation-based training and thus limits its availability to those times when staff are on hand.

Adaptive agents are needed to make simulation-based training more instructionally effective and available. This capability requires three developments. First, agents must be built using architectures capable of generating realistic and instructive behaviors. Second, data from simulators must be sufficiently rich to drive these agents. If the same data feeds are available from operational flight equipment, then agents used in training may eventually transition to the operational environment, to support pilot decisions and accelerate tactical execution. Third, a testbed is needed that can deliver some of these data and performance measures to support development of agents and evaluation of their tactical smarts and agility. In the best case, the data and measures will be so voluminous that developers can apply modern machine learning techniques to efficiently create adaptive agents as powerful as the current AI champions of Go [1] and many video games [2].

The Air Force Research Laboratory, 711th Human Performance Wing/RHA has invested in precisely these capabilities over four years, engaging eight of the leading developers of intelligent agents. In this paper, we describe key functionality of adaptive agents, and the data requirements these solutions impose on simulators and operational systems that drive them. We describe a testbed for agent development and evaluation that fulfills many of these data requirements, and concisely profile the agent architectures that testbed currently hosts.

2 Functions of Smart, Agile Agents

Four functions enable pilot agents to behave in a smart and agile manner: tactical inference, tactical action, modal behavior, and instructional capability. Two functions support maintenance and extension and use of smart agents: evolution and transparency. These functions can reside in agents themselves, or they can be distributed across an agent and the system that hosts it. Here, we define these six functions.

Tactical inference enables the agent to interpret instants and streams of tactical data (e.g., altitude, airspeed) in ways that support decision making. In Endsley's terminology of situation awareness [3], the agents must *perceive* the state of the environment, *comprehend* its tactical importance, and *project* (i.e., predict) potential evolutions of the environment.

Tactical action is the agent's capability to respond smartly to the tactical situation with actions that drive the adversary toward defeat and (if the adversary is a trainee) toward learning. An agent typically must synchronize its tactical actions with friendly forces to achieve strong and sure effects, and so the concept of tactical action necessarily includes communication and coordination between entities.

Modal behavior is the capacity of the agent to adopt distinct modes of tactical inference, tactical action, and other characteristics. For example, agent inference might be modeled on the doctrine of adversary force X in one mode and Y in another; an agent's tactical actions may vary on speed and synchronization as a function of the mode of expertise (e.g., novice, apprentice, journeyman, master); or an agent might vary its risk tolerance, and thus its tactical decisions, as a function of personality modes[1].

[1] Just how much variability in behavior is instructionally or operationally useful is an empirical question, as is the value of psychological realism (such as decision biases described by Tversky & Kahneman [12]) in agent processing.

Instructional capability is the ability of the agent to selectively produce tactical behaviors that exercise and develop the skills of trainee pilots. Such an agent must maintain a model of the trainee skill state, the target skill state (typically represented by training objectives or a model of expert behavior), and the degree to which the available tactical conditions exercise deficient trainee skills. In addition, a training agent should generate guidance and feedback for trainees, and if it can do so, communicate these effectively but selectively based on some instructional strategy.

Evolution is the ability of an agent to learn from new data, whether those data arise from training simulations, live operations, or other sources such as tactical simulations. Evolution requires careful management. An agent must evolve at a deliberate pace, meaning that the agent should learn or apply newly learned skills at a tempo that ensures a reliable training environment or operational capability for each cohort of human pilots who must fight from the same foundation of tactical knowledge.

Transparency is ability of an agent to describe the services it provides, the conditions under which it provides them, the accuracy and reliability with which it functions under those conditions, and the historical costs of modifying those services. With these descriptive and administrative metadata [4], an instructor or instructional system can select the optimal agent for a training task; and a user can estimate the return on investment of modifying an agent to provide new or better services.

By enumerating and defining these functions, we are better able to design adaptive instructional systems that can challenge trainees in a manner that is instructionally effective. Further, we are assured that these capabilities can be maintained and applied smartly and cost-effectively. In Fig. 1, we map the six functions, above, to three capabilities of an agent-based adaptive training system: domain proficiency, instructional efficacy, and maintainability and applicability.

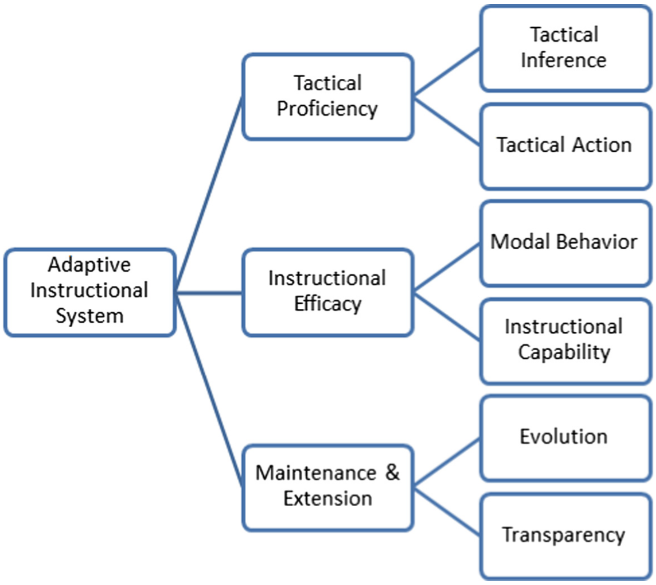

Fig. 1. Six functions (right) enable three capabilities (center) of adaptive instructional systems.

3 Data Requirements of Smart, Agile Agents

Agents can implement the functions, above, only if they have access to a range of data, which may be generated directly by agents, or by the host system on which they operate. Here, we define some of these data requirements, then briefly describe a testbed that delivers some of these data to agents developed by eight agent development firms.

3.1 Tactical Inference

Tactical inference implements situational awareness (SA). The three levels of SA defined by Endsley [3] impose distinct class requirements. To implement the *perceptual* level of SA requires a dynamic feed of data concerning the position, kinematics, and tactical actions of each entity in the battlespace (friendly, adversary, and neutral) and, in high fidelity simulations or operations, the state of the environment (e.g., visibility conditions ranging from dark to light, cloudy to clear).

To implement the *comprehension* level of SA requires the agent to infer missing data and transform perceptions into tactically meaningful constructs. The inference challenge is significant; an agent may need to infer missing dynamic data given incomplete or inaccurate persistent data. Entity and environmental data may not be available in their entirety to an agent either directly (e.g., through sensor readings) or indirectly, through reports by other entities such as an Airborne Warning and Control System (AWACS) or wingmen on a common network. Thus, an agent may need to infer attributes of an entity from incomplete data, as when a pilot infers that an aircraft is armed for battle given its speed, altitude, and heading. This inference requires persistent data concerning the potential order of battle, meaning the type, capability, number, and organization of entities in the battle space. Each of these may be complex data. For example, in tactical aviation, the capability of an aircraft includes at least its flight characteristics (e.g., potential speed, acceleration, and maneuverability), sensor capabilities, communications capability, and weaponry. In operations, these data may not be precisely known for any given platform, and it may not be known exactly which platforms the adversary will bring to a given battle. Training simulations typically evade the problem by providing agents with complete and accurate entity data, reducing the need for agents to implement rapid and accurate forms of inference. *Comprehension* SA also entails transformation of flight data into tactically meaningful constructs. The locations of entities in space must be transformed into the labels and parameters of formations documented in persistent data; the recent history of change in position and kinematics must be transformed into maneuvers defined in persistent data.

The *projection* level of SA requires the agent to predict how the battle will evolve. This, in turn, taps some persistent store of data concerning the responses of entities to potential actions of their partners and adversaries, and the effects of those actions on the mission. For example, an agent may accurately predict the maneuvers of enemy aircraft and estimate the threat those maneuvers pose to friendly units and their mission.

In sum, *tactical inference,* which implements situational awareness (perception, comprehension, and projection), requires persistent and dynamic data concerning the characteristics of entities and their behaviors.

3.2 Tactical Action

Tactical action entails selecting and dynamically adapting tactics to fulfill mission objectives and/or drive a trainee to learn. This requires persistent data concerning the effects that are appropriate given the mission (e.g., evade the enemy to approach a target, or engage the enemy to defend an asset), the tactics available, and the effects those tactics can achieve given the projected actions of the adversary. Persistent data concerning tactics must define formations, maneuvers, use of sensors and weapons, and, critically, coordination and communication with other members of the agent's force. Rich data supporting coordination might document a human or synthetic wingman's likely response time and accuracy of adherence to tactical doctrine and specific orders. Rich data supporting communication actions might document the effects of timing messages to arrive during periods of relative calm (when the recipient can attend to them) or relative chaos. Estimates of the effects of tactics may require relatively simple data (e.g., maneuver A has 75% likelihood of evading an adversary that is employing tactic B), or historical data that support a more game-theoretic decision strategy (e.g., with each repetition of tactic A, the likelihood of evasion drops by 50%).

These persistent data concerning planned effects, tactical options, and likely effects of tactical options in context may suffice to select tactics. To adapt tactical actions continuously to the moment requires the same dynamic data used in *tactical inference*.

3.3 Modal Behavior

Modal behavior by agents requires configuration data that specify the agent characteristics that an instructor (human or automated) or operator wants to evoke from an agent.

An agent's tactical behaviors might vary as a function of their configured nationality, where an agent representing a poorly trained and equipped adversary might select ill-fitting tactics predictably from a short playbook, while an agent representing a sophisticated adversary might select surprising and effective tactics from larger doctrine.

An agent's configured task work expertise might determine the accuracy and adaptability with which it executes a tactic at speed. Configurations for teamwork capability might drive the speed, accuracy, and timeliness of agent communications. Teamwork configuration data might control how well agents coordinate their actions to, for example, execute a pincer movement attacking an enemy's flanks.

A configuration of agent personality or attitude might address tolerance for risk, which in turn might bias the agent to select tactics that achieve effects in a manner that is more aggressive but potentially less survivable.

Implementation of modal behaviors requires persistent data that characterize the decision and behavior biases of different adversaries, at varied levels of expertise, or among different personalities. Data concerning the behavioral effects of these modes can be estimated from theory or research findings, or summarized from empirical data. In addition, parameter values must be available to select among these modes.

3.4 Instructional Capability

Instructional capability enables an agent to apply tactics that train. Such instructive behavior requires that an agent maintain a model of the trainee skill state relative to the target skill state. Measures of performance and effectiveness (MOPs and MOEs) populate trainee skill models; training objectives (or expert models) define the target skill states. Both types of data are necessary for instructional capability. They are not sufficient. To smartly select actions that train requires at least data that map training objectives to the training conditions that grow skill. For example, to develop pilot competence in tactical communication requires scenarios in which a wingman or AWACS operator communicates accurately (or error-fully) with the trainee. There may be many training conditions that exercise a given skill. It is not trivial to choose the most effective among them for a given trainee. Ideally, an agent's tactical actions invoke a training challenge that is neither too small to trigger human learning, nor too great to prevent it; rather, it falls in the Zone of Proximal Development [5] in which the agent can manage student learning. To adapt the challenge to the trainee requires data concerning the effects of the challenge on trainees' task work and teamwork skills given trainees' state (or state history). Note that trainee state may include physiological data that indicate attention, workload, and other conditions that bear on learning. Systems that capture physiological data should make measures of these conditions available to agents.

Human ability to learn from experience – from interaction with intelligent agents and other humans – is robust. However, these effects increase for some trainees when they are prepared for the specific learning experience, when they receive coaching (or otherwise "scaffolded") during it, and when they get feedback afterwards. Agents should train more adaptively and effectively if they can generate, communicate, and smartly deliver (or withhold) advanced organizers, real-time coaching, and debriefs based on instructional strategy. For example, immediate feedback benefits less proficient trainees, who are unable to assess their performance accurately, but may hamper journeymen, who must assess themselves in operational settings. Agents thus require data concerning the impact of instructional actions (not just tactical ones) on learning given trainee state.

3.5 Evolution

Evolution ensures that agents maintain or grow their tactical and instructional effectiveness as trainees, training objectives, measures, and the tactical environment change over time. Agent developers typically deliver evolution by manually modifying agent software. New machine learning techniques – notably deep learning from a static dataset and reinforcement learning from a dynamically generated dataset – are a cost-effective alternative that has produced superior agents in Go [1] and other domains [2]. Manual development benefits from, and automated learning requires, voluminous and variable data concerning tactical states (developed in tactical inference, above) and tactical actions, as well as measurements of their effects in battle, and ideally of their similarity to doctrinal tactics (procedures). These data can be captured in training simulations, simulated tactical exercises, and real operations for use by agent developers and machine learning.

Evolution requires careful management, as we note above. An agent that evolves more quickly than its users may appear to them to be unpredictable and untrustworthy. Agents or training systems should provide some control over the frequency with which agents learn or at which they apply what they learn. This implies a requirement for learning rate and/or learning application parameters.

3.6 Transparency

Transparency enables an agent to describe itself to its users, both human and automated. The data required to do this are what the National Information Standards Organization calls descriptive metadata and administrative metadata [4].

Descriptive metadata document the services that an agent provides (e.g., pilots an F-35 and communicate with F-35 pilots and AWACS), the conditions under which it provides them (e.g., air superiority missions and ground attack missions), and the reliability with which it functions under those conditions (e.g., mean time between failure of tactical engagements). These data enable a system that hosts many agents to select those that are best for a given training task.

Administrative metadata enable users to manage a resource. Here, the required data concern the cost of acquiring an agent if it is purchased per use or per user (as are some software-as-service applications), applying that agent (particularly if its installation and configuration are complex), and modifying it for new uses. These cost metadata are sometimes competition sensitive, and agent vendors may not wish to publish them, but purchasers know them and may be able to make them available on their own systems. Doing so enables a user or a system to perform return-on-investment (ROI) calculations that trade the effectiveness of an agent (documented in its descriptive metadata) against its cost.

3.7 Summary

We have enumerated a representative sample of the data required to deliver six useful functions of smart, agile agents for adaptive instructional systems. Such systems should be able to generate instructional challenges for trainees by virtue of tactical inference and tactical action functions. They should select among those challenges to optimize learning speed (holding the proficiency target constant) and/or proficiency (holding training time constant) by virtue of modal behavior and instructional capability). It should be straightforward to maintain and apply these agents because they evolve in tactical and instructional competence, and their utility and cost are documented in metadata.

We know of no single agent or host system that satisfies all these data requirements. However, one AFRL research program is systematically developing a testbed and agents that fulfill some of these data requirements. We describe those products below.

4 A Testbed for Developing and Evaluating Agile Agents

AFRL has developed a unique testbed for developing and evaluating smart, agile agents [6]. We call this testbed AGENT, the Agent Generation & Evaluation Networked Testbed (see Fig. 2). In general, the testbed is distinguished by (1) batch configuration and execution of scenarios in which agents fight against agile CGF and (2) automated performance measures that are (3) recorded in a common data store. These functions enable developers and testers to generate large volumes of variable performance and outcome data that sample a massive tactical space. Developers can access those data for machine learning of agent capabilities such as classification of tactical states from environmental data, and selection of tactical responses from behavioral and outcome data.

The testbed fulfills some of the data requirements described above.

To support tactical inference, the testbed reports the standard entity position and kinematics data available through the DIS protocol. It also describes the adversary formation and location, much as an AWACS operator would do for pilots in flight, through a special purpose protocol called m2DIS. Finally, it responds to requests for certain fundamental tactical information at the comprehension level of SA, such as “Am I being threatened? Am I in the adversary’s weapons engagement zone? Where is my wingman in relation to me?” Agents in the testbed are responsible for predicting the evolution of the engagement.

Agents hosted on this testbed are responsible for the tactic selection and parameterization, the core function of tactical action as we’ve defined it above. The testbed enables agents to execute maneuvers by name (e.g., dogfight, posthole, drag) with parameters (e.g., terminal speed and altitude). Similarly, it enables agents to control sensors and weapons with fine granularity at an unclassified level. Testbed agents are responsible for coordinating actions between one another, such as ensuring that each agent is prosecuting a different adversary, or that both are converging on the same one.

To support modal behavior, the testbed employs a relatively expert enemy (to developers’ agile agents), which is implemented as behavior transition networks in the Next Generation Threat System [7]. We plan to eventually configure these to represent the multiple levels of tactical expertise of human trainees. Testbed developers are free to model and parameterize their agents to represent different nationalities, levels of expertise, and psychological profiles (such as risk tolerance).

To support instruction, the testbed computes measures of tactical procedures and effects. Agents can use those data to optimize their actions for instructional effects. The testbed also provides a graphical interface in which agents report or explain their actions. This feature has proved useful in assessing agents, and so we expect it to add value in After Action Reviews. The testbed currently does not issue other data specific to instructional decisions. However, members of the team have published [8] a specification for representing some of those data. Training Objective Packages represent training objectives, their relationships to scenario conditions, and the performance measures by which trainee skill on objectives is evaluated. TOPs are designed to enable

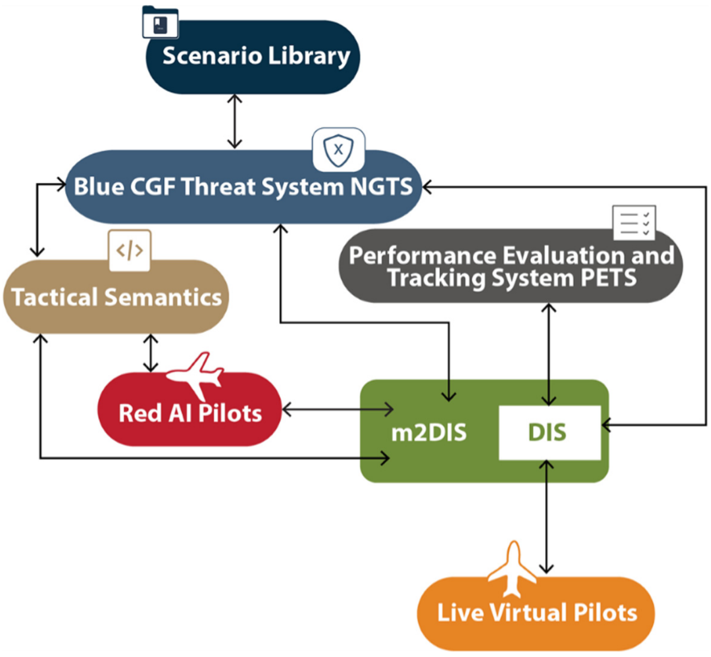
Scenario Library
Blue CGF Threat System NGTS
Tactical Semantics
Performance Evaluation and Tracking System PETS
Red AI Pilots
m2DIS
DIS
Live Virtual Pilots

Fig. 2. AGENT provides agent developers with a secure and private environment for exercising their red AI pilots against blue CGF in scenarios defined in a shared library. Blue airframes, sensors, and weapons are modelled in the Next Generation Threat System (developed initially by the Air Force and now by the Naval Air Warfare Center Training Systems Division). AGENT publishes standard entity state and interaction data using the DIS protocol. It computes tactically meaningful information concerning the tactical situation over a custom Model to DIS (m2DIS) interface. Measures of performance and effects are computed automatically by the PETS Performance Evaluation and Tracking System. Users can observe scenario runs on the LNCS LVC Network Control Suite (not shown). (Color figure online)

trainers or simulators to generate scenario events that bear on training objectives, and to measure the tactical and instructional effects. A future version of the testbed may incorporate TOPs.

To support evolution, the testbed computes measures and stores them in a shared data store from which any agent developer can draw data concerning the performance of every agent. These data can be voluminous because the testbed's batch operation function and agile opponent pilots. This should provide enough data for manual analysis and for machine learning.

To support transparency, the system will eventually document the capabilities of each agent in context. With these data, we will implement a function that selects the best agent for the training task from a library of agents.

5 Agent Architectures

The architectures and methods used to develop agents in this AFRL program are each capable of fulfilling some mix of the functions, above. Here, we briefly profile the architectures and call out just one or two agent functions each supports particularly well.

TiER1 Performance Solutions, LLC, is using a hybrid architecture that integrates two different human behavior representations. A task network model represents operator goals or functions in graph form. An accumulator model aggregates data over time to control transitions through the task network. Thus, this architecture can smoothly adapt tactical inferences and actions (i.e., decision making) in a dynamic tactical scenario.

Stottler Henke Associates, Inc., applies the SimBionic architecture [9], which implements an agent as an integrated set of behavior transition networks. This open source architecture supports a dynamic scripting machine learning algorithm, developed to adapt the behavior of agents by learning from experience. Thus, SimBionic has unusual strength with respect to agent evolution.

Soar Technology, Inc., applies the Soar cognitive architecture [10], a production system that searches a problem space and dynamically revises agent knowledge and actions to accomplish goals. If programmed at sufficiently fine level of granularity, a production system can effectively generate novel tactical inferences and actions. Thus, Soar agents are particularly capable of tactical inference and action within scenarios, and potentially of evolution over them.

Aptima, Inc., is applying deep learning techniques to infer tactical state and appropriate tactical response. These populate a Behavior Definition Language (BDL) that expresses goals, tactical state, behavioral constraints, actions, predictive measures and other attributes necessary for intelligent agent behavior. BDL is input to Soar agents. This work exemplifies automated evolution of agents.

Eduworks Corporation employs Brahms, a government-owned agent modeling framework created to design, simulate, and develop work systems, which consist of humans and technologies. Accurate representation of human-human and human-machine interaction make Brahms particularly capable of realistic coordination in tactical action [11].

Discovery Machine, Inc., applies a cognitive architecture called DMInd that represents hierarchies of pre-specified problem spaces and response strategies, which are retrieved as a function of fit to context. These functions are designed for accurate inference and action concerning tactics, but they can be applied to manage instruction as well.

CHI Systems applies its Personality-enabled Architecture for Cognition (PAC), a system that uses narrative threads to control perception and behavior. PAC explicitly represents personality and emotion. This makes it capable of modal behavior as a function of attributes such as risk tolerance and perception of threat.

Charles River Analytics, Inc., employs the Situation Assessment Model for Person-in-the-loop Evaluation (SAMPLE), which emulates recognition-primed decision-making, and the Hap model of reactive, goal-focused behavior. Thus, SAMPLE is well-suited to emulating expertise in tactical inference and action.

6 Conclusion

This article describes a set of functions that software agents should provide if they are to be tactically smart, instructionally effective, and cost-effective opponents in simulation-based flight training. Those functions, in brief, are tactical inference, tactical action, modal behavior, instructional capability, evolution, and transparency. These capabilities have interesting implications for data requirements, ranging from kinematic data already published by many simulators to training objectives and student skill assessments that are typically maintained only in the minds of expert instructors. A testbed developed by AFRL satisfies some of these new data requirements. Several agent development firms are testing the sufficiency of those data to drive smarter, more agile agents for training and, one day, for operations in battle.

Acknowledgements. This article is based upon work supported by the United States Air Force Research Laboratory, Warfighter Readiness Research Division 711 HPW/RHA, under Contract FA8650-16-C-6698. This article is cleared for public release on 28 Jan 2019, Case 88ABW-2019-0371.Thanks also to members of the following organizations for the concepts and agents they have developed in this program: Aptima, CHI Systems, Charles River Analytics, Discovery Machine, Eduworks, Soar Technology, Stottler Henke Associates, and Tier1 Performance Solutions.

References

1. Silver, D., et al.: Mastering the game of Go without human knowledge. Nature **550**(7676), 354–359 (2017)
2. Mnih, V., et al.: Human-level control through deep reinforcement learning. Nature **518** (7540), 529–533 (2015)
3. Endsley, M.R.: Toward a theory of situation awareness in dynamic systems. Hum. Factors **37**(1), 32–64 (1995)
4. Riley, J.: Understanding Metadata. National Information Standards Organization, Baltimore (2017)
5. Vygotsky, L.: Mind in Society: The Development of Higher Psychological Processes. Harvard University Press, MA (1978)
6. Freeman, J., Watz, E., Bennett, W.: A testbed for developing & evaluating AI pilots. In: Proceedings of ITEC 2019, Stockholmassan, Sweden (2019)
7. Next Generation Threat System. http://www.navair.navy.mil/nawctsd/pdf/2017-NGTS.pdf. Accessed 1 Jan 2017
8. Stacy, W., Freeman, J.: Training objective packages: a mechanism for enhancing the effectiveness of simulation-based training. Spec. Issue Theoret. Issues Ergon. Sci. **27**, 149–168 (2016)
9. SimBionic GitHub. https://github.com/StottlerHenkeAssociates/SimBionic. Accessed 21 Dec 2018
10. Laird, J.E.: The Soar Cognitive Architecture. MIT Press, Cambridge (2012)

11. Bell, B., Bennett, W., Clancey, W.: Socio-technical simulation for denied environments training: a contested airspace example. In: Proceedings of the Interservice/Industry Training, Simulation, and Education Conference (I/ITSEC). National Defense Industry Association, Arlington, VA (2018)
12. Tversky, A., Kahneman, D.: Judgment under uncertainty: heuristics and biases. Science **185**, 1124–1131 (1974)