Received: 5 June 2021 | Revised: 16 November 2021 | Accepted: 17 November 2021

DOI: 10.1002/ail2.54

PERSPECTIVE

WILEY

Explaining autonomous drones: An XAI journey

Mark Stefik[1] | **Michael Youngblood**[1] | **Peter Pirolli**[2] | **Christian Lebiere**[3] | **Robert Thomson**[4] | **Robert Price**[1] | **Lester D. Nelson**[1] | **Robert Krivacic**[1] | **Jacob Le**[1] | **Konstantinos Mitsopoulos**[3] | **Sterling Somers**[3] | **Joel Schooler**[2]

[1]Intelligent Systems Laboratory, PARC A Xerox Company, Palo Alto, California, USA

[2]The Institute for Human and Machine Cognition, Pensacola, Florida, USA

[3]Human-Computer Interaction Institute, School of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA

[4]Army Cyber Institute, United States Military Academy, New York, New York, USA

Correspondence
Mark Stefik, PARC, Intelligent Systems Laboratory, 3333 Coyote Hill Road, Palo Alto, CA. 94304.
Email: stefik@parc.com

Funding information
Defense Advanced Research Projects Agency, Grant/Award Number: FA8650-17-C-7710

Abstract

COGLE (*CO*mmon *G*round *L*earning and *E*xplanation) is an explainable artificial intelligence (XAI) system where autonomous drones deliver supplies to field units in mountainous areas. The mission risks vary with topography, flight decisions, and mission goals. The missions engage a human plus AI team where users determine which of two AI-controlled drones is better for each mission. This article reports on the technical approach and findings of the project and reflects on challenges that complex combinatorial problems present for users, machine learning, user studies, and the context of use for XAI systems. COGLE creates explanations in multiple modalities. Narrative "What" explanations compare what each drone does on a mission and "Why" based on drone competencies determined from experiments using counterfactuals. Visual "Where" explanations highlight risks on maps to help users to interpret flight plans. One branch of the research studied whether the explanations helped users to predict drone performance. In this branch, a model induction user study showed that *post-decision explanations* had only a small effect in teaching users to determine by themselves which drone is better for a mission. Subsequent reflection suggests that supporting human plus AI decision making with *pre-decision explanations* is a better context for benefiting from explanations on combinatorial tasks.

KEYWORDS

autonomous systems, explainable artificial intelligence, hierarchical multifunction modeling

Material within this technical publication is based upon the work supported by the Defense Advanced Research Projects Agency (DARPA) under contract FA8650-17-C-7710. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the official policy or position of the Department of Defense or the U.S. Government. Approved for Public Release, Distribution Unlimited.

1 | INTRODUCTION

COGLE (COmmon Ground Learning and Explanation) is an *e*xplainable *a*rtificial *i*ntelligence (XAI) system for autonomous drones that deliver supplies to field units in mountainous areas. Missions in COGLE take place in a simulated world with mountainous and forested settings, bodies of water, and structures. Figure 1 shows a mission map and a flight plan for an AI-controlled drone. The yellow stick figure shows where the hiker is. The curved arrow shows a drone's flight plan. The timeline below the map shows the *altitude* of the drone along its flight plan. Symbols on the maps indicate objects. The pointy symbols are high mountains that are too high to fly over. The curve-topped symbols are low and high foothills. Green areas are meadows. The tree-shaped symbols represent forests.

Initially, we used ArduPilot SITL,[1] which simulates low-level craft actions at high fidelity. The computational resources required for ArduPilot's detailed simulations proved unwieldy and unnecessary for the strategic planning of missions. Low-level flight control is widely implemented in commercial autopilots and hobbyist drones. To focus on mission planning, we developed a lower precision simulation model ("ArduPilot Light") with five levels of altitude and eight unique directions in a turn-based grid world. We modeled a compatible programming interface (API) of ArduPilot Light on the API of ArduPilot SITL. Figure 2 illustrates the coarse granularity of COGLE's simulation grid world for mission planning.

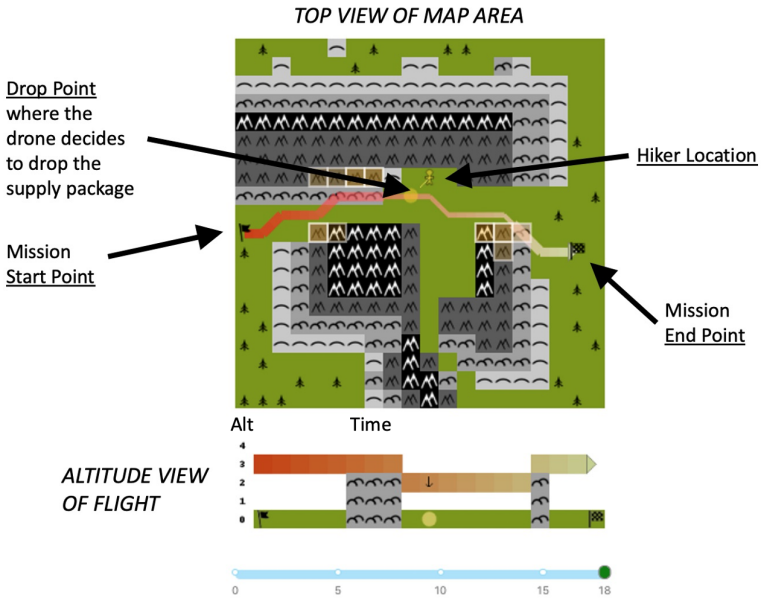

FIGURE 1 Example map for a mission in *CO*mmon *G*round *L*earning and *E*xplanation (COGLE's) domain

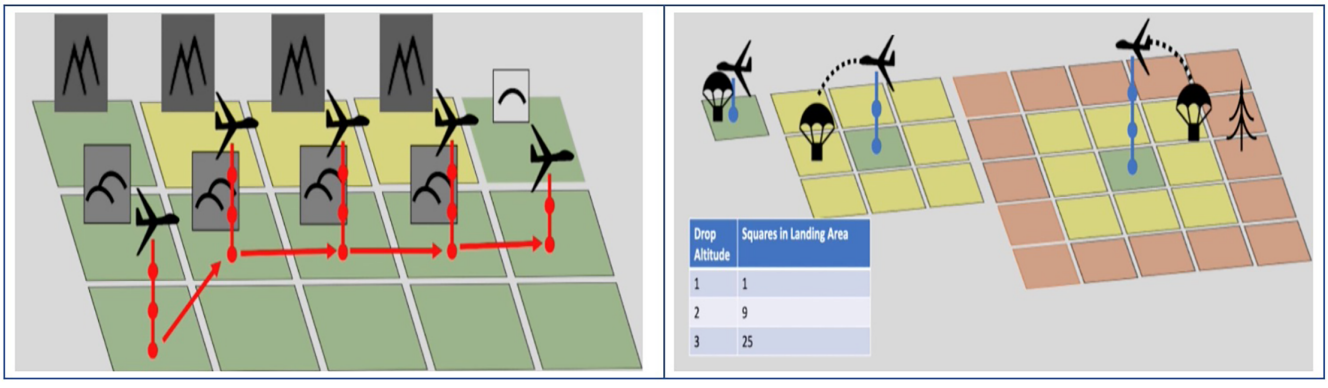

FIGURE 2 Illustrations from COGLE's flight school showing illustrating the model with five discrete altitudes and the increasing spread of the drop zone when packages are dropped from different altitudes

When drones fly too close to obstacles that are at the same altitude or higher, they risk crashing. If a drone releases its package above forests, high foothills, or water, then its package may be damaged. A package may become inaccessible landing in a river, trees, or high foothills. The higher a drone flies the further its packages may drift during the parachute drop. An AI pilot might take risks at the beginning, middle, or end of a mission. A pilot's early decisions on a mission can interact in subtle ways with later ones. For example, choices about how to avoid an obstacle early in a flight plan can lead to a route that precludes safe approaches to a chosen point to drop a package much later.

Using an early version of COGLE, we carried out self-explanation studies with users, as described[2] by Gary Klein, Robert Hoffman, and Shane Mueller among others. Such studies can provide a lens on the kinds of explanations that participants want and use for themselves. The AI pilots used for the drones were based on our early deep reinforcement learner (RL). They exhibited strange and suboptimal looping behaviors on very simple missions. The study participants cited observed patterns in the drone's behavior referring to inferred goals, utility, and drone preferences.

When asked to make predictions during the study, a frequent participant response was "I don't know." Study participants were creative in their self-explanations ("It is afraid of water!"), but they had no reliable basis for determining whether their interpretations were correct. It turned out that the strange behaviors of our early AI-controlled drones were due to their limited training.

2 | CREATING COMPETENT AI PILOTS

Training a drone to a high level of competence required architecting a neural network to generalize more effectively, generating enough earth-realistic training examples to cover the diverse topographical configurations of missions, and arranging training examples in a staged curriculum to facilitate machine learning.

The deep RL used a convolutional neural network (CNN).[3] Agents were trained using A2C,[4] a synchronous advantage actor critic. The deep RL architecture had two separate pipelines encoding two different image views of the drone's activity as shown in Figure 3. The allocentric view with two fully connected blue layers is used in the top pipeline. The egocentric view with two fully connected green layers is used in the bottom pipeline. The two pipelines are concatenated in a fully connected layer shown in purple. The network has two outputs: the expected reward/outcome $V(s_t)$ from a given state (ie, how much reward the drone should expect from current state), and the learned policy $\pi(a_t|s_t)$ that maps a state s_t to a probability distribution over actions. The approach described above is a model-free RL approach. The CNN and action-layers learn in concert. The whole network learns and encodes competencies to maximize reward.

CNNs are typically used to analyze visual imagery. CNNs support shift invariance, that is, they learn to detect visual features no matter where they appear in the image. With shift invariance, the CNN does not need more training examples to learn to recognize a feature everywhere it could appear.

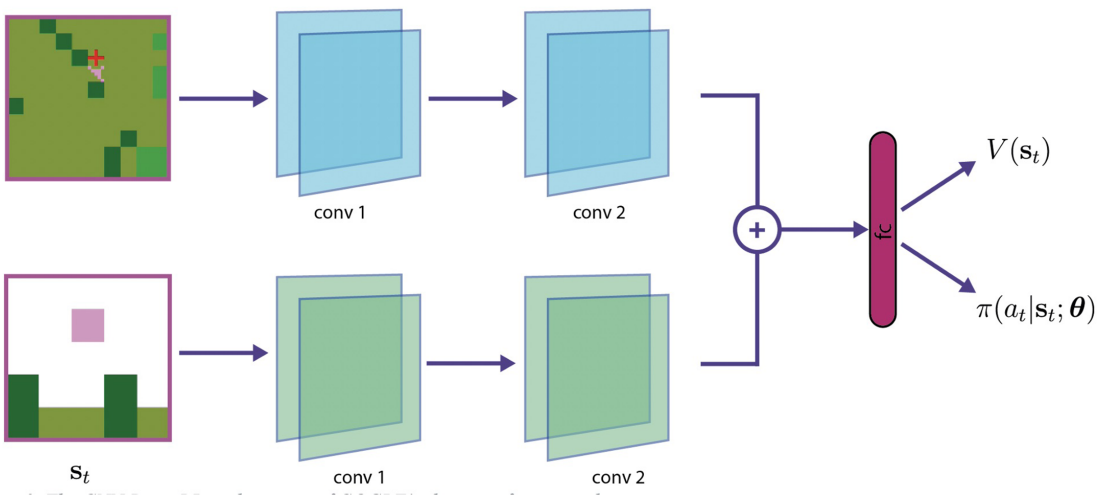

FIGURE 3 The architecture of *CO*mmon *G*round *L*earning and *E*xplanation (COGLE's) deep reinforcement learner

There are additional invariants and abstractions that people use in free space navigation. They learn these abstractions when they are very young and do not notice them later. Within the COGLE project, we referred to these abstractions and competencies informally as "toddler common sense" (TCS). Although such informal language is not typically part of the terminology of the machine learning research community, we found it useful for describing what the agents needed to learn.

Some of the TCS competencies about navigation in free space are suggested in the panels in Figure 4. The top row in panel (A) shows a craft flying east towards a hiker with no blocking obstacles. This case is easy because only two moves are required to go from the craft location to the hiker. Once a deep RL learns this case for flying east in the first row, it can handle a similar case in the second row because it has learned shift invariance and feature invariance. Typical CNNs also do not have rotational invariance. The case in the rightmost column flies the drone north rather than east. Flying north requires additional learning cases, as does flying in other directions including changes of altitude in a three-dimensional (3D) space. The craft in the bottom row flies east but requires more steps to reach the destination. The case shown in the right tile board requires the craft not only to reach a specific location, but also to arrive with a specific heading.

Panel (B) gives examples of navigation competencies for going over or around obstacles. An RL-based system needs to generalize its policy to apply to various kinds of obstacles, such as trees, watch towers, and so on. Panel (C) shows examples of competencies for choosing safe package release points. The bottom case in panel (C) is about penalizing the dropping of packages on the far side of a river that acts as a blocking object for the hiker. An AI pilot needs to be able to make successful plans for any arrangement of mountains, foothills, forests, water bodies, and hiker locations that can appear in a mission setting. Flight planning is sensitive to small changes in terrain. For example, adding an obstacle in a canyon or other tight space can make flying through it or turning around impossible or prevent descent to a low package dropping altitude, requiring routing changes to earlier parts of a plan.

We developed a procedural content generator to create diverse and realistic terrain examples. COGLE's procedural generator draws on technical approaches developed for movie and video game industries to generate realistic artificial and imagined landscapes. Roland Fischer et al. review[5] this technology. Figure 5 shows how COGLE's generator

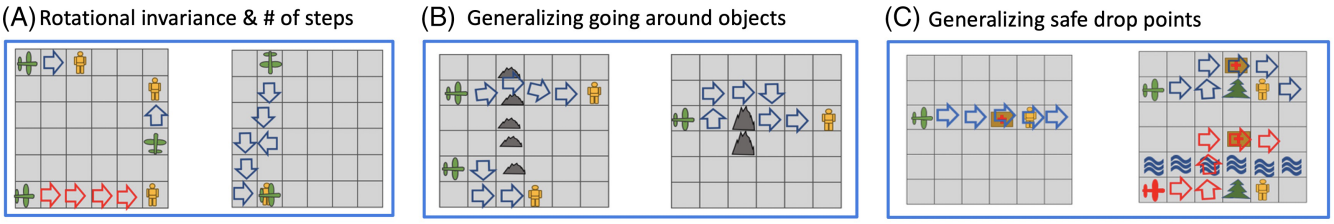

FIGURE 4 Example cases where changes to the convolutional neural network (CNN) architecture were needed to pick up abstractions beyond shift invariance

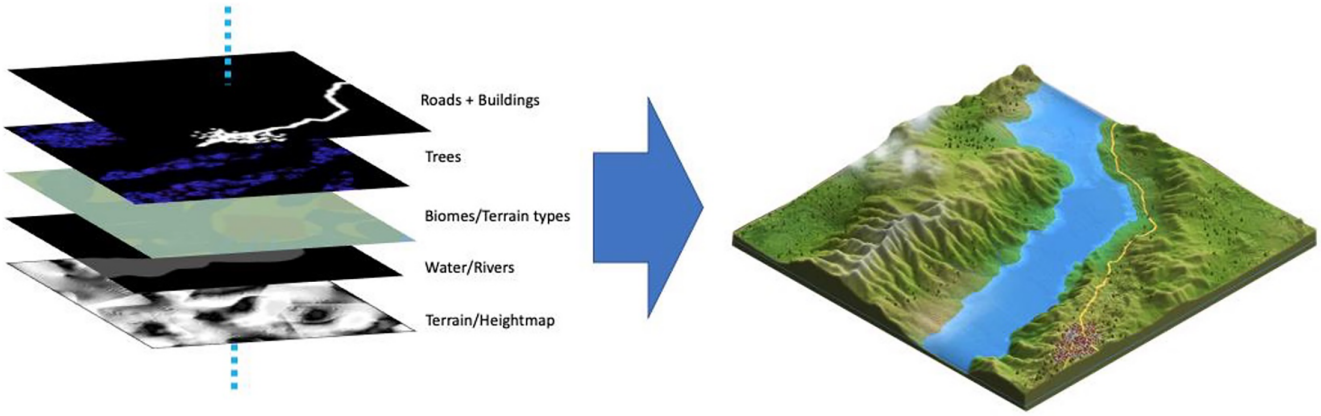

FIGURE 5 *CO*mmon *G*round *L*earning and *E*xplanation (COGLE's) procedural content generator used earth-realistic generative models with layers for terrain, rainfall, biome, and roads and buildings

created and composed layers representing terrain, rainfall and water bodies, biomes and terrain types, roads, and buildings.

The low-level competencies of our retooled machine learning approach are far removed from the concerns and interests of COGLE's intended end users and study participants. When our earlier self-explanation study participants tried to understand the strange behavior of the under-trained drones, they were used to reasoning about adults who possess mature navigational competencies. The low-level competencies needed to be acquired for the AI agents to be worth using.

3 | EXPERIMENTS WITH A COGNITIVE ARCHITECTURE

Like other model-free deep RL systems, COGLE represents what it learns in a distributed neural network. Causal explanations of an AI's behavior use symbolic representations. They are hierarchical and compositional. Producing such explanations requires bridging the gap from distributed representations to symbolic representations.

This section describes our experiments with a cognitive architecture[6] to bridge this gap. A cognitive architecture is a computational framework for representing cognitive processes including attention, memory, and pattern-matching, organized in a structure analogous to that of the human brain. We used the ACT-R architecture[7] developed by John Anderson and colleagues. ACT-R stands for Adaptive Control of Thought and the "R" stands for "Rational." Anderson developed ACT-R to explain how the brain works when it is learning. ACT-R is organized into modules for declarative memory, procedural knowledge, vision, audition, and motor control. In previous work[8] Lebiere et al. showed how an ACT-R cognitive architecture can replicate human sensemaking processes to transform loosely structured data from heterogeneous sources into a structured symbolic form.

Building on an approach that some of us had used previously to capture human reasoning, we set up drone missions to be annotated by human users using a domain ontology. COGLE agents would then plan the same missions. The next steps involved training a cognitive mapper and using the mapper to generate appropriate symbol structures for explanations. To train the mapper, we would analyze patterns of activations in the nodes in the upper levels of the neural network for missions, and then correlate those activation patterns with activation patterns in an ACT-R cognitive model of how humans solved the same missions. This training would provide a foundation for generating XAI explanations.

When we started developing this mapping approach, COGLE's learning system was not yet competent at planning drone missions. We opted to first develop a proof-of-concept system for the cognitive architecture approach using simple problems and missions in StarCraft 2 (SC2).[9] SC2 is a real-time strategy game where players control units from a third-person perspective with the aim of eliminating opponents. We used a system configuration for our experiments that included a neural network architecture like COGLE's. The focus was to understand what an analysis would tell us about the ascribed mental states of the learned model.

The analyses of our experiments and the sample problems in SC2 are detailed elsewhere.[10,11] While the research progress on analyzing the cognitive activity of the neural networks was encouraging, there were too many engineering tasks to use this approach for our COGLE end-user studies. We continued this research but developed an alternative explanation approach in parallel.

4 | FROM "INTROSPECTIVE" TO "EXTROSPECTIVE" EXPLANATIONS

The cognitive architecture approach to explanation is "introspective" in that it analyzes patterns of activity in the agents' neural networks. We pivoted to an alternative "extrospective" approach where the explainer employs the simulator to conduct experiments that analyze the *performance* of agents without inspecting their internal representations.

To create shared and precise language about COGLE's decisions and rationale, we adopted practices and terminology from formal games,[12] with game rules, outcomes, strategies, and strategic interdependence. In this approach, a game is any rule-governed situation with a well-defined outcome. Outcomes are game states that result from player actions. Strategy refers to a plan of action intended to lead to a desired terminal outcome. Strategic interdependence means that outcomes depend on the strategies used by the players.

COGLE's simulator has rules for actions and for rules for winning. Action rules govern how a drone, hiker, or package can move in the world model as depicted in Figures 1 and 2. The outcome rules determine changes of state in a turn, including changes to location, passage of time, and risk. The data describing the outcomes are stored in simulator

state variables. For example, local outcome rules define the increased risks for a drone when it flies too close to objects at the same or higher altitude. Additional state variables represent delays experienced by the hiker when a package lands out of sight or the hiker needs to climb a hill, walk around an obstacle, or swim in a body of water. Figure 6 shows the hierarchies of state variables representing risks and delay times. The rules for "winning" are that the drone whose plan incurs the lowest risk wins if it does not violate any time or risk constraint.

The panels in Figure 7 show block diagrams for two configurations of COGLE, with a simulator, an explainer, and AI pilots based on deep reinforcement learning or symbolic planning. The simulator models the world. AI pilots make mission decisions. The explainer analyzes flight plans and compares the pilots and their plans. Programmatic interfaces between the blocks enable a learning system to develop a policy, an AI pilot to take actions in the world, and an explainer to set up mission situations as experiments with the drones so that it can compare plans and create explanations.

Panel (A) shows the configuration with the deep RL. A learning phase precedes the human-machine task of choosing the better drone for a mission. The deep RL learner considers mission cases generated by the procedural content generator. It builds up from the TCS level concerns to task level concerns, incrementally improving the competence of an AI pilot based on feedback about outcomes in the simulated world. In the performance phase, the learned policy is fixed and functions as an AI pilot. Presented with a challenge mission, the AI pilot chooses a sequence of actions.

Panel (B) shows an analogous configuration using a planner. We used a variation of the A* search algorithm[13] with the provision that any path that violates a constraint is rejected. A* searches for an optimal solution for the given drone

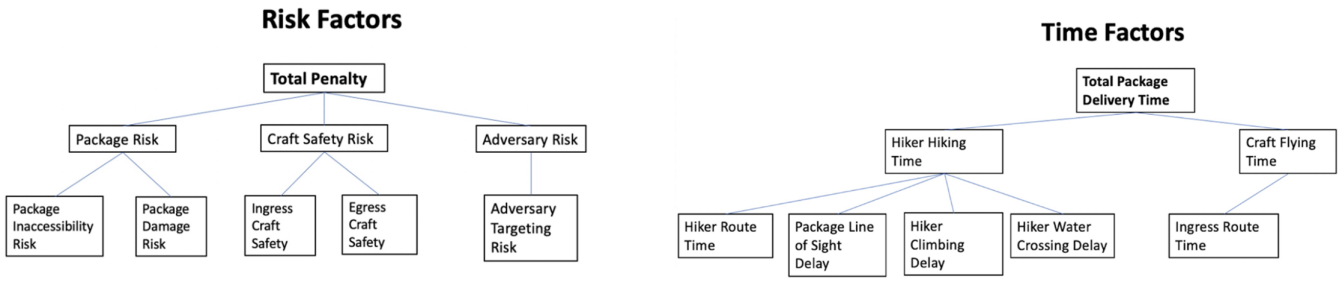

FIGURE 6 Accessible state variables in the simulator corresponded to factors for risk and time

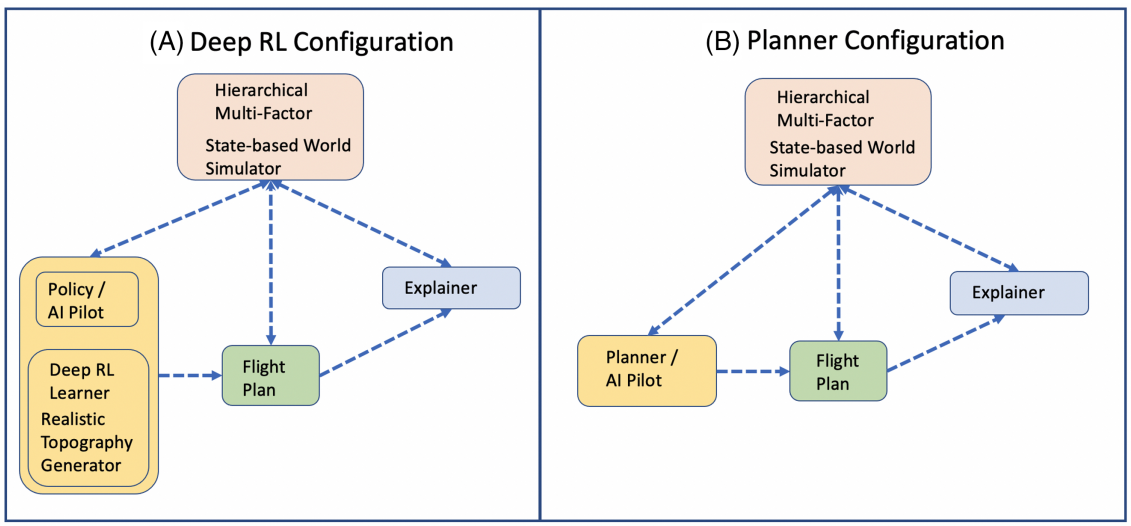

FIGURE 7 Block diagram showing two versions of *CO*mmon *G*round *L*earning and *E*xplanation (COGLE's) system architecture. One version uses learned policies to fly the drone. The other uses a variation of a simple A* planner. In both configurations, the explainer produces explanations by comparing two drone performances in the simulated world

on the given mission. In our implementation, this computation typically takes about 20 minutes, making this the more convenient configuration for experimenting with game rules without days of training.

The simulator provides a set of terms with consistent meanings that can be used in explanations to users. An example of a narrative explanation comparing the flight plans of two agents is:

> "Drone 1 drops the package where there is lower package inaccessibility risk, because only it can safely fly close to obstacles."

The term "lower package inaccessibility risk" corresponds to the aggregated state variable "package inaccessibility risk." It is computed by the simulator as part of the outcome state representing where a package landed. Similarly, observations about drone performance such as "only it can safely fly close to obstacles" (or paraphrased "it incurs lower risks when flying close to obstacles") reference the state variable "craft safety risk."

Martin Erwig et al. describe[14] a method for decomposing rewards or penalties into different types and explaining choices of actions by a minimal sufficient explanation (MSE). An MSE is a minimal set of rewards that are sufficient for distinguishing the benefit of one action over another. Extending this work, we developed a hierarchical multi-factor framework (HMF) for decision problems. The HMF framework aggregates factors. For example, one drone may incur greater risks to craft safety to reach a drop point where a package can be released to land with lower inaccessibility risk, lower package safety risk, or lower package delivery time. The different kinds of risks such as package risk or craft safety risk correspond to the types of rewards or penalties described in the paper by Erwig et al. but extended to have type hierarchies. Risk factors quantitatively relate to probabilities of damage that imperil the success of a mission. Time factors correspond to the time it takes a drone or hiker to reach a destination. In HMF, the conditions for winning are described by a combination of constraints on state variables and optimization criteria. In COGLE, the drone with the lowest total risk penalty wins unless it violates a time constraint.

Figure 8 shows COGLE's narrative and visual explanations that are intended to help users to understand the drones and their plans. The explanation system creates narrative explanations using terminology from the HMF hierarchies. The narrative explanations explain advantages for *what* the drones did and competency differences as reasons for *why* the drones could do it.

COGLE's computation of the narrative explanations combines MSEs with counterfactual cases. The "*what*" phrase in the narrative explanation is the factor with the biggest risk difference. To simplify the narrative, sibling factors low

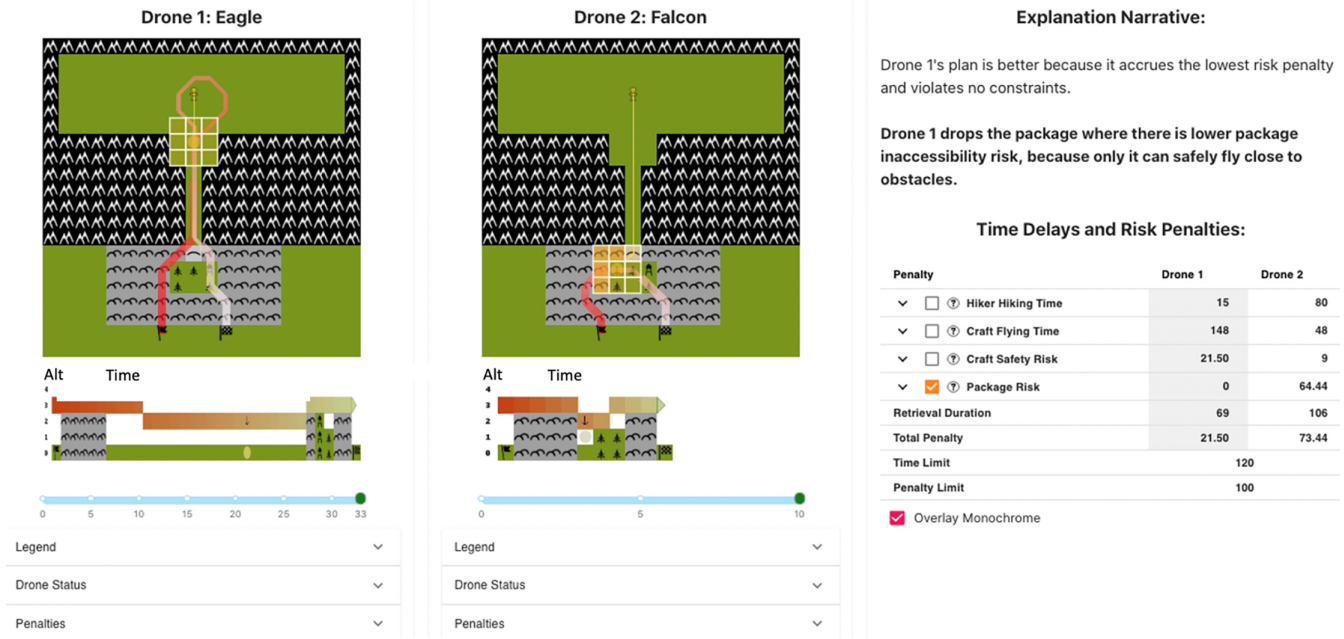

FIGURE 8 Example mission with explanations with two candidate drones and their different flight plans. Narrative explanations say what the winning drone did and why it had an advantage. The interactive visual explanation shows where on the map the crafts encountered risks. As a user scrubs through the timeline, obstacles on the map light up with increasing intensity indicating greater risk

in the hierarchy can be combined and accounted for in terms of their parent risk unless the contribution is mostly from one low-level risk. For example, multiple low-level package accessibility factors can be aggregated to the parent "package accessibility" factor.

Consider the earlier explanation example,

> "Drone 1 drops the package where there is lower package inaccessibility risk, because only it can safely fly close to obstacles."

The "what" phrase about lower package accessibility risk identifies which action resulted in the greatest risk difference. The explanation generator assigns credit to a single factor if it is responsible for a clear majority of the risk difference. When the risk is aggregated from several contributing sibling risks in the factor hierarchy, it simplifies the explanation by assigning credit to their parent risk. In this example, the explanation generator associates the action of dropping a package at a location with the factor for lower package accessibility risk. When a drone wins because the other drone violates a constraint on a factor, the competency associated with that factor is assigned the credit.

Producing the "*why*" phrase requires collecting counterfactual data using the simulator to conduct experiments. The explainer instructs the simulator to fly the losing drone using the flight plan of the winning drone. Low-level craft flight differences show up as different amounts of risks for the two drones taking the same actions. In this example, the losing drone (Falcon) flying Eagle's plan would incur excessive craft risk. This is "why" it did not choose Eagle's plan and why it was forced to choose an inferior location to drop the package.

To help users understand these flight differences and risks in the context of the mission map, the narrative explanations are augmented by visual explanations. The visual explanations overlay the map with highlights to call attention to places *where* risks occur. The risky obstacles are identified and displayed on the map by the simulator. As a participant scrubs the timeline, the drones move, and the simulator highlights the risks accumulated so far in the flight plan. Drones can also win because of superior information processing. Differences in information processing competencies show up as differences in state variables representing the information that the drone uses.

To test the effectiveness of our XAI explanations we set up a model induction user task as described in Appendix. The weak model induction improvements with *post-decision explanations* motivated us to pivot to a decision support context with *pre-decision explanations*.

5 | SUMMARY AND REFLECTIONS

One way that AIs can have value is by carrying out important cognitive tasks better than people. Like other multi-factor optimization problems, the nature of COGLE's mission planning is combinatorial. During the XAI project, we came to recognize that combinatorial problems create challenges for people, challenges for training deep reinforcement learners, challenges for using cognitive models to generate XAI explanations, and challenges for carrying out model induction user studies.

Stepping back again, our experience recalled earlier AI examples where humans and machines solve combinatorial puzzles. Based on heuristic search, the DENDRAL program[15] routinely solved structure elucidation problems on mass spectroscopy data. Carl Djerassi[16] reported on a graduate chemistry class that he taught at Stanford University where he assigned students to use DENDRAL to check the correctness of chemistry papers that had been published in scientific journals. They found that practically every published article missed solutions or reported incorrect ones. This showed that even in professional scientific publications, AIs could find errors that were missed by expert authors and reviewers.

Another analogous and widely reported example of AI superior performance is move 37 in the Go match between AlphaGo and Lee Sedol. The rules of Go are well known. The combinatorial challenge is to thoroughly explore possible solutions. As described in an article[17] in *Nature*, AlphaGo reached a "superhuman" level in the strategy game Go, without learning from any human moves. How even world-class human players were surprised by the winning line of play has been analyzed and discussed for several years.

These observations suggest framing the teaming of people with AIs in decision support tasks that exploit human and AI strengths. The role of the AI can be to efficiently solve given problems and the role of humans can be to oversee and adjust the problem criteria. The XAI explanations help people to check the reasonableness of AI solutions. By analogy, it is easier to check a proof than to create one. In contrast to the *post-decision explanations* required for a model induction study, *pre-decision explanations* support human understanding of AI reasoning in a context of decision support. Pre-decision explanations enable a person to check an AI's solution, even when finding the optimal solution is difficult for an unaided person to

do routinely. Analogous divisions of labor across human plus computer teams have been studied previously in analytic tools for sensemaking based on interactive information visualization.[18]

Appendix. Model Induction User Study

User study participants are presented with a mission and two different AI-controlled drones as shown in Figure 9. The drones perceive their environment differently, perform differently, and make different flight plans. Participants are asked to choose the best drone for each mission. After participants pick a drone, the system shows COGLE's explanations.

The structure of our model induction user study is shown in Figure 10. Our hypothesis was that the users receiving XAI explanations after choosing the best drone for a mission would learn to make better choices about the best drone in later missions. The study task missions were arranged in a curated curriculum that introduced domain challenges incrementally. Initially, participants were exposed to situations where one drone's plan was better for a single main reason. In later missions, the terrains were more complex and the reasons for a plan's advantage varied. Missions were randomly ordered as to which drone had the advantage.

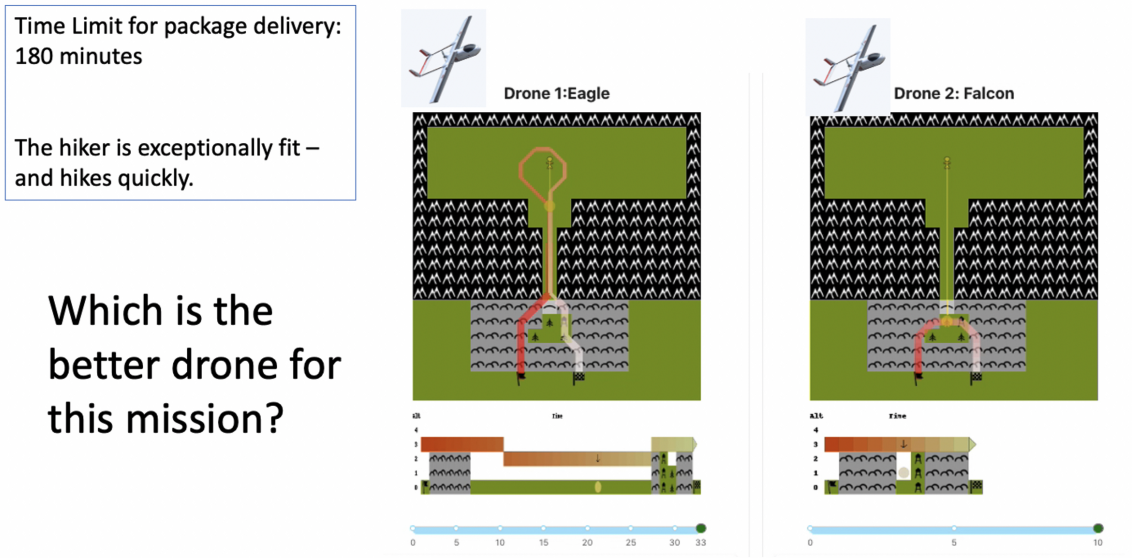

FIGURE 9 Users are presented with two drones and their flight plans for a mission. They are asked to select the best drone for a mission

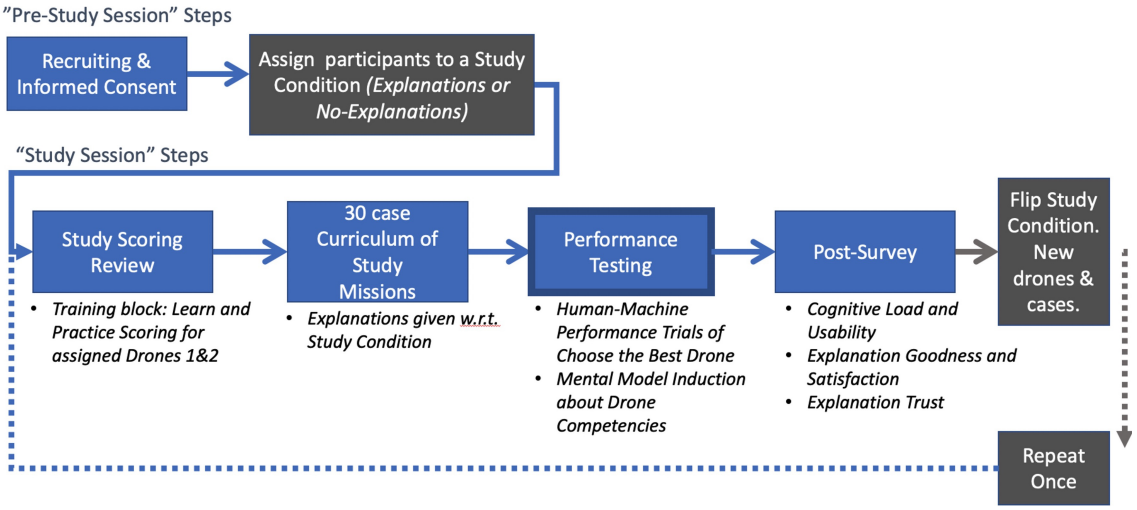

FIGURE 10 Study plan for the user study. In each case, users were asked to select the best drone for a mission and afterwards were shown the correct answer and given the explainable artificial intelligence (XAI) explanations. We removed the qualification step in the final study because flight school did not improve user performance in predicting the performance of the drones

N = 92 participants were recruited for this study from Amazon Mechanical Turk. Of these N = 92, N = 76 successfully completed the study (17% excluded from analysis for incomplete responses). Participants were randomly assigned to either receive explanation first or explanation second. Furthermore, participants were also randomly assigned one of the pairs of drones to see first.

We used a 2 × 2 within-participants (repeated measures) design with a within-participant variable of explanation interface (explanation, no-explanation) and a between-participants variable of order (explanation first, no-explanation first) and counterbalanced with Drone Pairs (Eagle and Falcon First, Osprey, and Peregrin First).

Each participant proceeded through a training block followed by two testing blocks. In a training block, we gave participants a description of and training on the domain world and how to select which drone would win a mission. Approximately, half the participants received the explanation interface in testing block 1 and then the no-explanation in testing block 2 at random. The remaining participants had the opposite order (no-explanation in testing block 1 followed by explanation in testing block 2). Each testing block consisted of 30 mission trials. In each testing block trial, participants were presented with two different drones on the screen. The first 10 mission trials were designed to have participants experience a successive revelation of drone behavior as described above. Of those 10 missions, the first four, Eagle or Osprey, would win. The following two missions were won by Falcon or Peregrin. Then the missions would vary winning and losing. From missions 11 to 30 within a testing block, the terrains were arranged randomly so that different drones had the advantage.

The primary objective of this experiment was to determine whether more people correctly predicted the better of two drone flights in the explanation condition than in the no-explanation conditions. We defined success rate to be the proportion of correct predictions over 30 trials in the explanation or no-explanation condition. We ran a paired *t*-test testing the difference between the explanation conditions success rate ($M = .57$, $SD = .16$) and no-explanation's rate (M = .54, SD = .16). The mean of the differences (0.04) suggests that the effect is positive, significant, and small (95% CI [1.78e−03, 0.08], $t\,(75) = 2.08$, $P < .05$; Cohen's $d = 0.24$, 95% CI [0.01, 0.47]).

Figure 11 shows that there was a small positive model induction effect of explanations as measured by the participants ability to correctly predict which drone was better for the mission. The explanation condition produced a small but reliable effect on the ability to predict future drone success. It is specific to the drones explained.

Figure 12 illustrates the performance of the participants over the aggregated test conditions using a logistic histogram as described[19] by Jennifer Smart and colleagues. The red line is the fit of a logistic regression to the percentage of accurate predictions. The case numbers refer to different missions and drones in the balanced conditions of the study. Participants in both conditions improved in performance as they trained in more missions. The difference between the explanation and no-explanation conditions contributes approximately as much benefit as one additional case in the 30 cases of each condition.

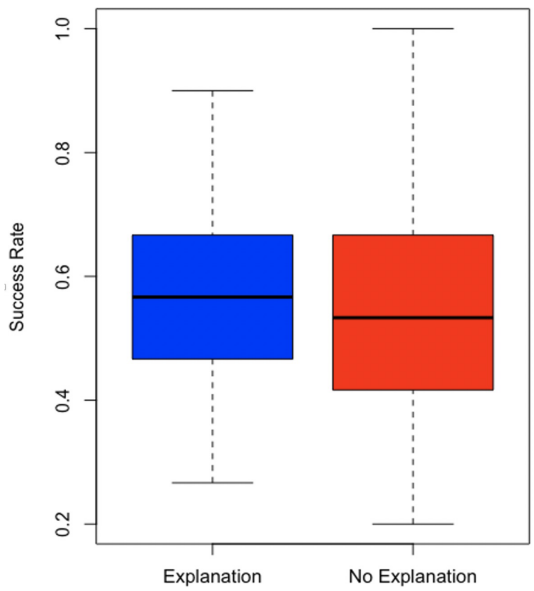

FIGURE 11 Boxplot showing the explanation and no-explanation success rates (proportion of correct predictions for 30 trials in each condition)

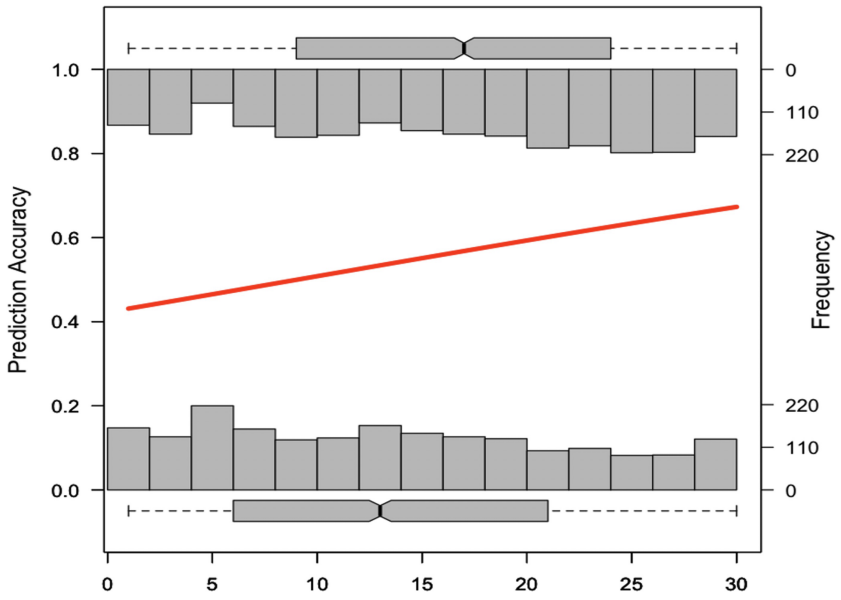

FIGURE 12 Logistic histogram plot showing prediction accuracy by trial as well as frequency histograms of successes (1) and failures (0) by trial. The y-axis for the histograms is on the right. Top box plot is the distribution of success over trial; the bottom box plot is the distribution of failure over trial. The boxplots can be interpreted in the usual manner: the central notch is the median, the gray box encompasses the second and third quartiles, and the outer whiskers are the extremes (1.5*inter-quartile range). The red line is the fit of a logistic regression. The y-axis for the fit line is on the left

The model induction user study showed that the effect of the after-decision explanations was weak in improving the performance of participant prediction on future missions.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author upon reasonable request.

ORCID

Mark Stefik https://orcid.org/0000-0003-3134-6031
Michael Youngblood https://orcid.org/0000-0001-9490-0015
Peter Pirolli https://orcid.org/0000-0002-9018-4880
Christian Lebiere https://orcid.org/0000-0003-4865-3062
Lester D. Nelson https://orcid.org/0000-0001-8556-7644
Konstantinos Mitsopoulos https://orcid.org/0000-0002-0076-2334
Sterling Somers https://orcid.org/0000-0003-0129-7080

REFERENCES

1. ArduPilot Development Team. ArduPilot SITL (Software in the Loop) Simulator. https://ardupilot.org/dev/docs/sitl-simulator-software-in-the-loop.html
2. Klein, G, Hoffman, R, Mueller, S.T. Naturalistic psychological model of explanatory reasoning: how people explain things to others and to themselves. *Conference on Naturalistic Decision Making*, San Francisco, 2019.
3. Le Cun Y, Boser B, Denker JS, et al. Handwritten digit recognition with a back-propagation network. In: Touretzky D, ed. *Advances in Neural Information Processing Systems 2*. Vol 89. Denver, CO: Morgan Kaufman, NIPS; 1990:1990.
4. Mnih, V., Badia, A.P., Mirza, M., Graves, A. Lillicrap, T.P., Harley, T., Silver, D., Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. International Conference on Machine Learning 1928–1937, ICML, 2016.
5. Fischer R, Dittmann P, Weller R, Zachmann G. AutoBiomes: procedural generation of multi-biome landscapes. *The Visual Computer*. 2020;36:2263-2272.
6. Kotseruba I, Tsotsos JK. 40 years of cognitive architectures: Core cognitive abilities and practical applications. *Artificial Intelligence Rev*. 2020;53:17-94.
7. Anderson JR. *Rules of the Mind*. New York: Psychology Press. Lawrence Erlbaum Associates; 1993.

8. Best BJ, Gerhart N, Lebiere C. Extracting the ontological structure of OpenCyc for reuse and portability of cognitive models. *Proceedings of the Behavioral Representation in Modeling and Simulations (BRIMS) Conference*. Charleston, SC; 2010.
9. Vinyals O, Ewalds T, Bartunov S, Georgiev P, Vezhnevets AS, Yeo M. Starcraft ii: a new challenge for reinforcement learning. *arXiv* preprint arXiv:1708.04782. 2017.
10. Somers S, Mitsopoulos K, Thomson R, Lebiere C. Explaining decisions of a deep reinforcement learner with a cognitive architecture. *Proceedings of the 16th International Conference on Cognitive Modeling*. Montreal, QC, Canada: ICCM; 2018:144-149.
11. Mitsopoulos K, Somers S, Thomson R, Lebiere C. Cognitive architectures for introspecting deep reinforcement learning agents. *Workshop on Bridging AI and Cognitive Science, at the 8th International Conference on Learning Representations*. ICLR; 2020.
12. Gardner R. *Games for Business and Economics*. John Wiley and Sons; 2003.
13. Hart PE, Nilsson NJ, Raphael B. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transact Syst Sci Cybernet*. 1967;4(2):100-107.
14. Erwig M, Fern A, Murali M, Koul A. Explaining deep adaptive programs via reward decomposition. *International Joint Conference on Artificial Intelligence (IJCAI) Workshop on Explainable Artificial Intelligence*. 2018;40-44:100-107. https://www.ijcai.org/proceedings/2018/
15. Lindsay R, Buchanan B, Feigenbaum E, Lederberg J. DENDRAL: a case study of the first expert system for scientific hypothesis formation. *Artif Intell*. 1993;61(2):209-261.
16. Djerassi C. *The Pill, Pygmy Chimps, and Degas' Horse*. Lexington, Massachusetts: Basic Books; 1993.
17. Gibney E. Self-taught AI is best yet at strategy game go. *Nature*. 2017.
18. Heer J, Viégas FB, Wattenberg M. Voyagers and voyeurs: supporting asynchronous collaborative information visualization. *Communications of the ACM*. 2009;52:1.
19. Smart J, Sutherland WJ, Watkinson AR, Gill JA. A new means of presenting the results of logistic regression. *Bull Ecol Soc Am*. 2004;85(3):100-102.

How to cite this article: Stefik M, Youngblood M, Pirolli P, et al. Explaining autonomous drones: An XAI journey. *Applied AI Letters*. 2021;2(4):e54. doi:10.1002/ail2.54