Collaborative Autonomy through Analogical Comic Graphs

Matthew Klenk[1], Shiwali Mohan[1], Johan de Kleer[1], Daniel G. Bobrow[1], Tom Hinrichs[2] and Ken Forbus[2]

[1]Palo Alto Research Center, 3333 Coyote Hill Road, Palo Alto, CA, 94304

[2]Northwestern University, 633 Clark St, Evanston, IL 60208

{klenk,mohan,dekleer,bobrow}@parc.com and {hinrichs,forbus}@northwestern.edu

Abstract

For more effective collaboration, users and autonomous systems should interact naturally. We propose that sketch-based interaction coupled with qualitative representations and analogy provides a natural interface for users and systems. We introduce *comic graphs* that capture tasks in terms of the temporal dynamics of the spatial configurations of relevant objects. This paper demonstrates, through a strategy simulation example, how these models could be learned by demonstration, transferred to new situations, and enable explanations.

Introduction

While there have been tremendous achievements in machine learning (e.g., AlphaGo [10]), significant challenges remain for the widespread adoption of autonomous agents in open world mission critical applications. First, while games enable straightforward definitions of goals and rewards, many applications require complex tradeoffs over varying timescales. Second, in open worlds, the training and deployment environments frequently differ. Third, autonomous systems do not work in isolation, but as teams; therefore, they must be able to explain their actions to facilitate trust and improve team performance.

As observed by inverse reinforcement learning [9], agent designers may only have a rough idea of task reward functions. Natural collaboration cannot start with users providing reward functions. Instead, our approach learns *inspectable models* from example executions. Inspectable models use relational representations that provide human-like similarity inferences, via analogy, ensuring that the agent's decisions make sense to users. Example task executions could be abstracted into an inspectable model that captures the spatial temporal relationships between relevant objects and regions as indicated through a sketch-based interface. We call this model a *comic graph,* and, in this paper, we describe how comic graphs could be used to learn new tasks, to transfer acquired knowledge to new situations, and to collaborate with users by explaining the agent's decisions.

Comic graphs are qualitative representations of spatial configurations through time that describe events and tasks. Execution histories are segmented into snapshots, like panels in a comic strip, where each panel represents a distinct state with temporal relations between them (e.g., Fred exiting a room would have three panels: (1) Fred walking toward the door, (2) Fred in the doorway, and (3) Fred outside the door). This sequence of comic panels both explains the exiting room task as well as providing guidance for an autonomous agent performing the task. Many tasks are performed in multiple ways (e.g., Fred may stop to pick up his backpack before walking out the door), therefore each panel may have multiple successors in the comic graph. Previous research has shown these qualitative relational representations enable event detection and explanation in video [2]. Here we show how these representations support training, performance and explanation in an unmanned aircraft (UAV) attack mission.

Our approach is built on the Companion cognitive architecture [5], which is aimed at reaching human-level AI by creating software social agents (i.e., systems that interact with people using natural modalities, working and learning over extended periods of time as apprentices rather than tools). The two central hypotheses of the architecture and our approach are (1) analogical reasoning and learning are central to cognition, and (2) relational representations, especially qualitative representations, are key to human intelligence. The architecture is built on three computational models of analogical processes: matching (SME [4]), retrieval (MAC/FAC [3]), and generalization (SAGE [7]). For this paper, it is necessary to know that these processes operate over relational representations and have been used as cognitive models to account for experimental results.

Learn Comic Graph by Example

Consider training an agent to pilot a UAV on an attack mission. The UAV can move and fire its weapons. The target is a Surface to Air Missile site (SAM) that could shoot down the UAV. Users train a Companion by performing the missions and highlighting task relevant regions. The attack mission could be described in terms of four tasks: ingress, strike, egress and abort. While interacting with a simulator to control the UAV, the user communicates which task they are executing using natural language, and then through a sketching interface (CogSketch [6]), the trainer annotates the task-relevant objects and regions. This type of training requires considerably less expertise than fine tuning reward functions and can be potentially distributed across multiple trainers.

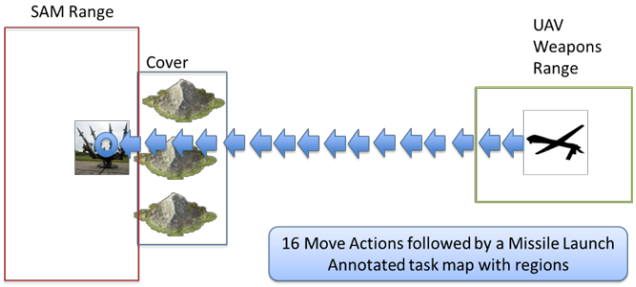

Figure 1: Training episode where the user specifies that they are demonstrating a strike task and indicates that the weapon ranges of the SAM and UAV are important as well as the fact that mountains provide cover.

Figure 1 illustrates a training episode for the strike task of an attack mission where the user takes 17 actions in the simulator and indicates three task relevant regions (the units' weapons ranges and cover) by sketching. After the SAM is destroyed, the user states that they have completed the strike task successfully and that the next task is to egress.

The Companion will abstract the spatial temporal configuration of these regions and objects into a sequence of comic panels that capture the changes in the qualitative relationships over time. In this case, there are four comic panels (summarized here in natural language): (CP_{11}) the UAV is approaching cover and target; (CP_{12}) the UAV continues approaching the target while moving through cover; (CP_{13}) the UAV and SAM are within weapons range of one another; (CP_{14}) the SAM is destroyed. As indicated above, the qualitative representations determine how the task execution is segmented temporally. This single sequence is a comic graph which serves as an inspectable model of how to execute a strike task during an attack mission. As the trainer performs additional strike missions, each resulting sequence of comic panels is merged into the comic graph via analogical generalization. Figure 2 illustrates how two additional examples are incorporated.

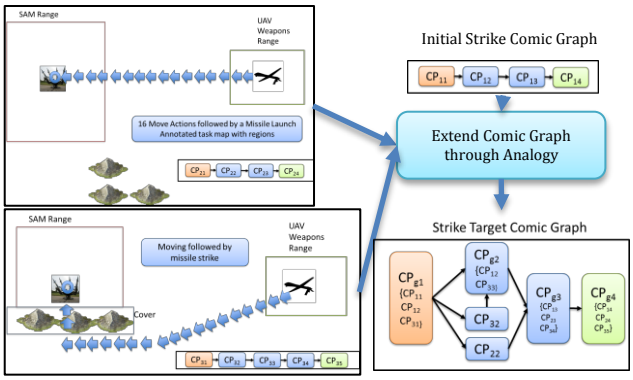

Figure 2: Comic graph (lower right) from multiple training instances. Orange comic panels correspond to initial task states and green panels represent terminal states.

Note that each training episode could occur not only over different configurations, but also different sets of relevant objects and regions. For example, in one of the episodes there is no cover region. The resulting comic graph (shown in Figure 2) captures different ways in which the strike task could be performed. From the initial state (a generalization of all three initial comic panels) where the UAV moves toward the target, it can either enter the UAV's weapon range (CP_{22}), fly parallel to cover (CP_{32}), or enter cover directly (CP_{g2}). After exiting cover, the UAV and SAM are in range of each other (CP_{g3}). In the final state (CP_{g4}), the SAM has been destroyed. The generalization panels (CP_{g1}, CP_{g2}, CP_{g3}, CP_{g4}) are generalizations created by SAGE capturing the commonalities between the examples while still maintaining links to the specific comic panels. Here there is only a single initial and terminal panel, but this representation allows multiple initial and terminal conditions. For example, if in a later training exercise, an opposing aircraft engaged the UAV, the trainer may elect to terminate the strike task and begin an abort task, flying away from the attack. This would create a terminal comic panel with a different next task transition than CP_{g4}.

To capture the fact that some ways of performing the task are better than others, we learn a value function over the comic panels. The value function captures how likely it is that the task will be executed correctly given the current comic panel. Successful execution is defined by arriving in a state that corresponds to a successful terminal comic panels. In the case of the training example without cover, the

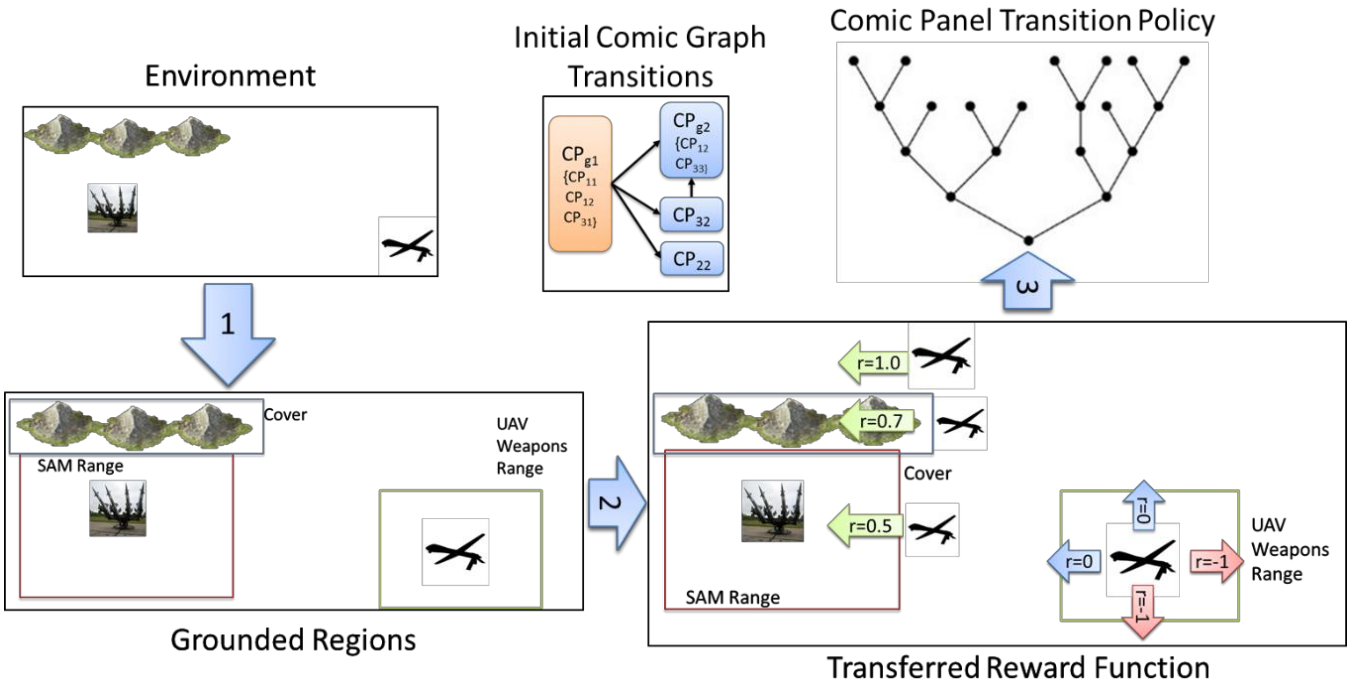

Figure 3: Comic graphs could enable robust autonomous action by (1) grounding relevant regions through analogy, (2) transferring the reward function from the comic panels to state transitions, and (3) directing policy search.

UAV may be shot down before striking the target resulting in a failed task execution. Consequently, CP_{21} and CP_{22} have lower value functions.

Ground Comic Graph into a Policy

Because the comic graph captures the spatial dynamics of successful task execution, it guides action selection in new environments. This is done in three steps: (1) using analogy, we ground the most similar initial comic panel to identify the objects and regions relevant to task execution; (2) through directed policy search, we identify a value function over the state action space; (3) using the learned policy, we select actions until a comic panel transition occurs. These steps repeat until the Companion enters a terminal comic panel that specifies the next task, and if there is a terminal panel with no next task, the mission is complete.

Figure 3 illustrates these three steps for the first comic panel transition of the learned strike task model. In the first step, using symbolic descriptions of spatial concepts [7], we identify the task relevant regions in the current state through analogical inferences from the initial task comic panel, CP_{g1}. Next, we ground the reward function and perform directed policy search to identify the policy that leads to the best comic panel transition. Using the simulator as a transition function $T(s,a) \rightarrow s'$, we perform Monte Carlo Policy Search [11] from the initial state, s_0. After each action, we determine the comic panel corresponding to the transition.

Consider the move left action, $T(s_0,a_{left}) \rightarrow s_1$, the state pair (s_0,s_1) corresponds to the comic panel CP_{g1} because they contain the same qualitative spatial relationships (the UAV is moving toward cover as well as the target and none of the regions are overlapping). Therefore, this reward is neutral. Now consider the move right action, $T(s_0,a_{right}) \rightarrow s_2$,. While the regions are still not overlapping, the objects are all moving further away. These relationships do not correspond to any comic panels of the current comic graph transitions. Therefore, we impose costs on these actions through a negative reward value. This process continues until we reach transitions that correspond to the successor comic panels $\{CP_{g2}, CP_{22}, CP_{32}\}$ or a finite horizon. By only looking ahead to single panel transition, this is an orders of magnitude smaller search space than performing a Monte Carlo Search for the entire task or mission. The transitions that result in the next comic panels are assigned positive reward and are marked as terminal reinforcement learning states.

We set the rewards for each comic panel by scaling its value function with the analogical similarity between the panel and the current situation. While the comic panels CP_{32} and CP_{g2} have the same value, they are scaled differently in the current situation because the current situation is more similar, as defined by SME, to the CP_{32}.

The directed search results in a policy tree that the Companion follows until the transition to CP_{32}. From the resulting state, the entire process iterates with the comic graph

transitions consisting of only $CP_{32} \rightarrow CP_{g2}$. This process continues until either there are no applicable actions (e.g., the UAV is destroyed) or the Companion reaches a terminal state in the comic graph. In this case, once the UAV destroys the SAM, the Companion enters the terminal panel, CP_{g4}, indicating the next task is egress. Next, the Companion executes the comic graph model of the egress task in the same way, and completes the mission when there are no more tasks to execute.

Breadth of Explanations

By capturing task relevant qualitative distinctions, comic graphs support a broad range of explanations. Through multi-modal interaction, the user should immediately recognize if the Companion is not considering task relevant objects or regions. Due to the user's understanding of the scenario, they should be able to quickly identify if an action is consistent with the Companion's expected comic panel transition. By showing the sequence of comic panels the Companion expects to satisfy the current task, the user could determine if that is a realistic plan for the current environment. Due the structure of comic graphs, the Companion can answer questions about the relationships between tasks and missions.

Comic graphs support the following types of questions. "What is the mission?" can be explained in terms of the task labels provided by the user during training. For example, the attack mission was learned in terms of four tasks with temporal relationships between them: ingress, strike, egress and abort. One level down, the user can ask "why is the current task 'strike?" Here the Companion will display the grounding of the terminal comic panel of the ingress task at the appropriate point in the execution history. Next, the user can ask, "what is the strike task?" which the Companion answers by displaying the comic graph of spatial configurations of relevant objects demonstrating the temporal sequence. User can also ask "why did you just take the move-left action?" Instead of trying to explain this in terms of the underlying value function, the comic graph highlights the qualitative transitions that the Companion is pursuing.

Another advantage of the comic graph structure is that it maintains example training and execution sessions. This allows for the following types of queries. The user can also ask "what is cover? And why is it relevant?" Here the system would produce previous examples where cover was used in destroy-target missions. Or when asking one of the above questions about the mission, task or spatial configurations, the Companion could produce previous examples that demonstrate the concept.

Explanations not only support joint execution performance, but also provide an opportunity for corrective feedback. This could be in terms of what parts of the scenario are relevant (e.g., "This is not cover") or about which action should have been taken (e.g., "You should move left here"). We use model based diagnosis [1] to scan back through the audit trails to determine which inference, analogy, actions, or representational choices contributed to the poor performance. The diagnoser can identify multiple possible corrective action sets to can repair the model (yet preserve prior learning). The diagnoser can determine the likelihood of each such set, and, if multiple alternative sets are equal likely to propose a question to the user.

Discussion

This paper introduces comic graphs as an inspectable models based on qualitative relations and analogy supporting human machine collaboration. Our approach uses comic graphs to address three autonomy challenges. Instead of hand authoring reward functions, comic graphs are learned by example through natural interaction. Second, comic graphs support flexible execution in new environments and open worlds through a combination of analogy and reinforcement learning. Finally, comic graphs enable explanations that improve individual and joint performance.

References

[1] Console, L., Hamscher, W. and de Kleer, J. (1989). Readings in model-based diagnosis, Morgan Kauffman.

[2] Dubba, K., Reddy, S., and Cohn, T. (2015). Learning relational event models from video. *Journal of Artificial Intelligence Research* 53: 41-90.

[3] Forbus, K., Gentner, D., and Law, K. (1995). MAC/FAC: A model of similarity-based retrieval. *Cognitive Science*, 19, 141-205.

[4] Forbus, K. D., Ferguson, R. W., Lovett, A., and Gentner, D. (2016). Extending SME to handle large-scale cognitive modeling. *Cognitive Science*, DOI: 10.1111/cogs.12377, pp 1-50.

[5] Forbus, K, Klenk, M. and Hinrichs, T. (2009, July/August). Companion Cognitive Systems: Design Goals and Lessons Learned So Far. *IEEE Intelligent Systems*, 24(4), 36-46.

[6] Forbus, K., Usher, J., Lovett, A., Lockwood, K., & Wetzel, J. (2011). CogSketch: Sketch understanding for Cognitive Science Research and for Education. *Topics in Cognitive Science*. 3(4), pp 648-666.

[7] Hawes, N., Klenk, M., Lockwood, K., Horn, G. S., & Kelleher, J. D. (2012). Towards a Cognitive System that Can Recognize Spatial Regions Based on Context. In AAAI.

[8] McLure, M., Friedman, S., and Forbus, K. (2010). Learning concepts from sketches via analogical generalization and near-misses. *Proceedings of the 32nd Annual Conference of the Cognitive Science Society*. Portland, OR.

[9] Ng, Andrew Y., and Stuart J. Russell. "Algorithms for inverse reinforcement learning." *ICML*. 2000.

[10] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G. & Dieleman, S. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587), 484-489.

[11] Sutton, R. S., & Barto, A. G. (1998) *Introduction to reinforcement learning* (Vol. 135). Cambridge: MIT Press.