Explainable Deep Reinforcement Learning for Aircraft Separation Assurance

Wei Guo
Department of Computer Science
George Washington University
Washington, DC 20052, USA
Email: weiguo@gwu.edu

Peng Wei
Department of Mechanical and
Aerospace Engineering
George Washington University
Washington, DC 20052, USA
Email: pwei@gwu.edu

***Abstract*—Aircraft separation assurance is an extremely challenging task in a complex airspace. Deep Reinforcement Learning (DRL) was used to develop aircraft separation assurance models in our previous works. Though these models have shown promising performance, the DRL agents make decisions in a nontransparent way, limiting their use in safety-critical applications. In order to build a trustworthy DRL model for aircraft separation assurance, we propose a novel framework to provide stepwise explanations of agent behaviors. At a high level, our framework distills a complex DRL model into a shallow Soft Decision Tree (SDT) and uses the distilled knowledge in SDT to provide visual explanations of agent behaviors in each step. Specifically, the proposed framework incorporates (1) a distillation module to transfer knowledge from DRL policies to tree-structured policies with clear decision paths and (2) a visualization module with a graphical interface to provide visual explanations in real time. With our proposed framework, the information is extracted from the distilled SDT and illustrated with the interface. Through extensive numerical experiments in an open-source air traffic simulator with challenging environment settings, our results show that the proposed framework can support explainable decision-making for aircraft separation assurance.**

I. Introduction

Deep Reinforcement Learning (DRL) has shown great success in sequential decision-making problems with high-dimensional and complex problem settings recently [1], [2]. Aircraft separation assurance system uses DRL to assure aircraft safety with complex and dense traffic [3]–[5]. The authors have developed the state-of-the-art multi-agent DRL methods for aircraft separation assurance in structured airspace [6]–[9]. These DRL algorithms are trained to learn compact representations of high-dimensional aircraft states to make tactical decisions. Though these methods have shown promising performance in aircraft separation assurance, existing DRL agents make decisions in a nontransparent way, selecting actions without providing explanations. The lack of transparency creates major obstacles to build trust in the decision-making process. This problem greatly limits the use of DRL algorithms in safety-critical applications, especially when human is in the loop.

To address the issue of nontransparency in DRL models, Explainable Deep Reinforcement Learning (XDRL) methods are proposed in the literature to provide explanations for agent actions [10], [11]. One popular XDRL approach is the Soft Decision Tree (SDT) [12], which is a combination of network distillation and decision tree models.

SDT has a similar tree structure to the Hard Decision Tree (HDT) [13]. While HDTs suffer from the fact that each node only relies on one feature, each decision node in an SDT is a one-layer neural network using all the features as input. SDTs were originally developed for image classification. Since the DRL models provide the states and actions as training features and labels, SDTs can be used to replicate the DRL policies in the supervised learning paradigm and provide insights into understanding agent behaviors.

Despite showing great potential in explaining the agent behaviors, current works related to SDT in DRL mainly focus on tasks with the low-dimensional input such as CartPole and tasks with the pixel-based input such as Mario AI Benchmark [14], [15]. These implementations rely on either low dimensionality or spatial patterns of the input, which greatly restricts its use in other fields with high-dimensional and non-spatial state space such as separation assurance.

In order to build a trustworthy DRL model for aircraft separation assurance task, we propose a novel framework to provide stepwise explanations of agent behaviors. In contrast to the related papers, our proposed framework can provide explainable decision-making in the complex separation assurance task, which helps build trust for this safety-critical system. At a high level, our framework distills a DRL model into a shallow SDT and uses distilled knowledge in SDT to provide visual explanations of agent behaviors stepwise. Specifically, the proposed framework incorporates (1) a distillation module to transfer knowledge from complex DRL policies to tree-structured policies with clear decision paths and (2) a visualization module with a graphical user interface to provide visual explanations in real time. By combining them together, the integrated framework can explain agent behaviors by (1) extracting information from the distilled SDT about feature importance and feature affections on actions and (2) illustrating the extracted information with the interface. We refer to our framework as "**S**tepwise **E**xplainable **S**eparation **A**ssurance **ME**thod" (for short SESAME).

Our main contributions can be summarized as follows:

- We propose a novel framework to provide explainable DRL for aircraft separation assurance task. The framework can help user understand the agent decision-making in real time.
- We introduce two original visualization methods namely tree plot and trajectory plot to demonstrate the extracted knowledge from tree-structured policies. The methods can help provide visual explanations efficiently.

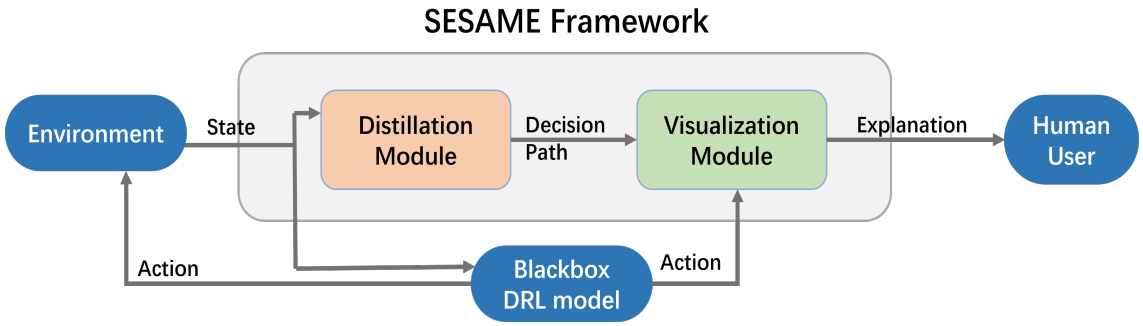


Fig. 1: The structure of the SESAME explanation framework. The framework observes the environment and generates the decision path with distillation module at each time step. The decision path is fed to the visualization module. The visualization module generates tree plot and trajectory plot to provide explanations of agent behaviors.

- We conduct extensive experiments to demonstrate the effectiveness of our proposed framework for providing decision explanations in aircraft separation assurance.

II. Related Work

DRL has been recently explored in aircraft separation assurance. In the last few years, the authors were among the first to develop DRL models for aircraft separation assurance [6]–[9]. Proximal Policy Optimization (PPO) [16] is one of the most popular models in separation assurance because of its high and stable performance (*e.g.*, [5]–[9], [17]). Deep Q-network [18] and its variants [19] are also widely used to maintain the safety separation (*e.g.*, [20]–[23]). In continuous action space, Deep Deterministic Policy Gradient (DDPG) [24] algorithm is broadly applied to resolve conflicts between aircraft (*e.g.*, [25]–[28]). However, these works train DRL model as a black box without behavior explanations, which limits users' trust in these models and further restricts their use in real-world safety-critical systems. In this work, we present a novel explanation framework to provide explainable decision-making for aircraft separation assurance. To our knowledge, our work is the first to provide behavior explanations in DRL models for aircraft separation assurance.

Past works on XDRL have explored various methods to derive explanations for model behaviors. Representation learning is applied to generate compact and explainable representations of agent in [10], [11]. Logic rule methods are employed to extract behavior explanations with human-readable rules from DRL model in [29]. Neural language models are trained to generate text explanations for agent behaviors in [30], [31]. Saliency maps can explain agent behaviors by highlighting the input features most relevant to the decisions in [32]–[34]. Reward decomposition method reformulates the reward function in DRL model to provide behavior explanations with meaningful and decomposed rewards in [35], [36]. Uncertainty estimation methods are applied to quantitatively measure the decision confidence and provide insights into the agent behaviors in [37], [38].

SDT is a promising approach to behavior explanations. SDTs can distill the knowledge from complex DRL models into shallow trees with decision paths, providing both clear interpretations and comparable performances to original DRL models. SDT was originally proposed for image classification task in [12]. Since the DRL models provide the states and actions as training features and labels, SDTs can be used to replicate the DRL policies in the supervised learning paradigm. Coppens et al. [15] distilled a SDT from a PPO network to explain behaviors of the agent playing Super Mario game [39]. Liu et al. [40] approximated Q function with the Linear Model U-tree to provide explanations. Silva et al. [41] discretized the SDTs with univariate nodes to further improve the interpretability. Ding et al. [14] combined a feature learning tree with standard SDT to allow rich expressivity for explanations. Dahlin et al. [42] proposed a collection of metrics for evaluating the distilled SDTs.

Our work is fundamentally different from the above papers in the following ways: (1) Our work focuses on aircraft separation assurance with a complex high-dimensional input space while the previous works on SDT mostly focus on the tasks with simple low-dimensional input (e.g., CartPole [14], LunarLander [41]) or tasks with pixel-based input (e.g., Mario AI Benchmark [15], Wildfire Tracking [43]). These implementations depend on the low dimensionality or spatial information of the input but cannot handle the tasks with a complex and non-spatial state space. (2) The aircraft separation assurance problem is formulated as a multi-agent reinforcement learning problem. Each aircraft is treated as an agent, which greatly increases the complexity of explanations since both interactions with the environment and each other need to be explained. But the previous works concentrate on single-agent problem settings [14], [15]. (3) We propose two visualization methods namely tree plot and trajectory plot to demonstrate both detailed and precise explanations efficiently based on extracted information from SDTs, while the related works only illustrate the explanations with heatmaps [15] or decision tree plots [43].

The remaining parts of the paper are organized as follows. Section III introduces the background. In section IV, the description of the problem is presented. Section V introduces our solution to the problem. Section VI provides the details

of the experiments and results. Section VII summarizes our findings and concludes the paper.

III. Background

A. Reinforcement Learning

Reinforcement Learning (RL) addresses the sequential decision-making problem and its objective is to learn the optimal policy within an environment. A Markov Decision Process is used to model it with tuple $< \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma >$:

- The state space $\mathcal{S}$ is the set of all states in environment.
- The action space $\mathcal{A}$ is the set of all actions that the agents can select.
- The state transition function $\mathcal{T}$ represents the probability of going from one state to another.
- The reward function $\mathcal{R}$ determines the amount of reward obtained by the agent given the state-action pair (s_t, a_t) at time step t.
- The discount factor $\gamma \in [0, 1]$ determines the importance of rewards in the future. A small γ prioritizes the immediate rewards and a large γ highlights the future.

The agent interacts with the environment by observing the current state s_t and selecting the action a_t based on its policy π_t at time step t. Then the agent receives a reward r_t and the environment produces an updated state s_{t+1}. The goal is to learn the optimal policy π^* which maximizes the cumulative reward. The optimal policy can be defined as:

$$\pi^* = \underset{\pi}{\operatorname{argmax}}\, E[\sum_{t=0}^{T}(r(s_t, a_t)|\pi)],$$

where T represents the total time, π represents any policy, and $r(s_t, a_t)$ represents the reward given pair (s_t, a_t).

B. Soft Decision Tree

Soft Decision Tree (SDT) [12] is a classification model which integrates the traditional decision tree with perceptrons. Each non-leaf node k of SDT is a single-layer neural network with weight parameter w_k and bias parameter b_k. Each leaf node l learns a vector parameter ϕ^l to form a prediction distribution over all possible classes as in:

$$Q^l_c = \frac{exp(\phi^l_c)}{\sum_{c'} exp(\phi^l_{c'})},$$

where c' is a possible class and c is the target class.

Given input state s and node k, the *traversal probability* defined as the probability of traversing to the left child node of node k is calculated as:

$$p_k(s) = \sigma(\beta(sw_k + b_k)),$$

where σ is the sigmoid function and β is the inverse temperature parameter.

The traversal from root node to another node (e.g., a leaf node) composes a *decision path*. The decision path shows hierarchical decisions made by the SDT and can provide behavior explanations. Each decision path has a certain *path probability* P which is the overall product of all probabilities leading from root to the last node in this traversal.

The loss function consists of the entropy loss for classification and a regularization loss to penalize unequal usage among nodes. The entropy loss calculates the entropy between prediction distribution Q and target distribution T in each leaf node l, weighted by its path probability $P^l(s)$:

$$L(s) = -log(\sum_{l \in \mathcal{L}} P^l(s) \sum_c T_c log Q^l_c),$$

where $\mathcal{L}$ is the set of all leaf nodes in the tree.

The regularization loss C is the cross-entropy between the current distribution α_k and the desired distribution α'_k of all non-leaf nodes in the tree:

$$C = \sum_{k \in \mathcal{K}} -\lambda_k(\alpha'_k log(\alpha_k) + (1 - \alpha'_k) log(1 - \alpha_k)),$$

where

$$\alpha_k = \frac{\sum_s P^k(s) p_k(s)}{\sum_s P^k(s)}.$$

Here $\mathcal{K}$ is the set of all non-leaf nodes. $P^k(s)$ is the path probability from root node to node k. $p_k(s)$ is the traversal probability of k. λ is a hyper-parameter to determine the strength of the penalty.

IV. Problem Formulation

In en route sectors, aircraft separation assurance is performed among all aircraft to maintain safety separation. Our proposed framework is used to provide behavior explanations for all aircraft. In this work, we use BlueSky [44] as the air traffic simulator. We evaluate the performance of proposed explanation framework on several challenging case studies with multiple intersections and high-density air traffic.

The objective of the case studies is to provide explanations of decisions to maintain safe separation between aircraft and resolve conflicts for all aircraft in the sectors. The proposed explanation framework needs to understand both the ownship information and the coordination information of other potential intruders to provide correct explanations.

Three different case studies are investigated in this work. Each use case is a dynamic simulation environment where the aircraft enter the sector stochastically. The setting further increases the difficulty of behavior explanations because now the framework needs to understand the strategies based on the varying environment information in real time.

Multi-Agent Reinforcement Learning Formulation. We formulate the aircraft separation assurance problem as a deep multi-agent reinforcement learning problem by treating each aircraft as an agent. The state space, action space, terminal states, and reward function are defined as follows.

1) State Space: The state contains all the information an agent has for decision-making. The functions of communication and coordination between agents are implemented in our work because of our cooperative multi-agent setting. The state information contains the information of both ownship and intruders in the sector. Specifically, the following features

are included: current location, speed, acceleration, route identifier, distance to sector exit, distance between ownship and intruders, and distance to intersections.

2) Action Space: Action space contains all actions the ownship can select. Agent is allowed to select action every 12 seconds because radar surveillance updates en route position every 12 seconds. Three actions are provided in action space:

$$\mathcal{A}_t = [a_-, a_0, a_+].$$

Here a_- refers to deceleration, a_0 refers to maintaining the current speed, and a_+ refers to acceleration.

3) Terminal State: In each simulation, aircraft will be generated in the sector until the maximum allowed number of aircraft is reached. The simulation will terminate when all aircraft have reached their individual terminal state, obtained in two ways: (1) exiting the sector without conflict, (2) violating the loss of separation with conflict.

4) Reward Function: Identical reward function is defined for all agents to encourage the cooperation. The system penalizes the conflict with a negative reward. The penalty is local so only the two or multiple aircraft in conflict will get penalized but the other agents will stay unaffected. Specifically, a conflict is defined that the distance between two aircraft is less than 3 nautical miles. The reward function for conflict is defined as follows:

$$R_c(s) = \begin{cases} -1 & \text{if } d_o^c < 3, \\ -\alpha + \delta \cdot d_o^c & \text{if } 3 \leq d_o^c < 10, \\ 0 & \text{otherwise.} \end{cases}$$

Here d_o^c is the distance from the ownship to the closet intruder in nautical miles. α and δ are scale parameters which ensure the reward is between -1 and 0.

We also introduce the penalty of speed changes, which should be avoided unless necessary in the real world:

$$R_s(a) = \begin{cases} 0 & \text{if } a = a_0, \\ -\psi & \text{otherwise,} \end{cases}$$

where ψ is a hyper-parameter implemented to minimize the number of speed changes.

$R_c(s)$ and $R_s(a)$ together guide the agent to learn a policy whose goal is to maintain safety separation with a small number of speed changes. We capture our goals in the following reward function:

$$R(s, a) = R_c(s) + R_s(a).$$

V. Solution Approach

Our objective is to provide explanations of the DRL agent behaviors in aircraft separation assurance. In order to achieve this goal, we propose an explanation framework to pair with the given DRL model. The framework consists of two components, namely, the *distillation module* and *visualization module*. The distillation module is a SDT transferring knowledge from complex DRL policies to tree-structured policies with clear decision paths. The visualization module provides visual explanations with a graphical user interface efficiently. We describe two modules in the following subsections and then show how they are integrated together as the SESAME framework. Finally, we discuss three techniques to efficiently improve the performance of proposed framework.

A. Distillation Module

This module distills a DRL model into a SDT. The DRL model controls the agent to perform aircraft separation assurance task. A dataset containing transitions generated by the DRL model is used to train the SDT model. The dataset consists of state-action pairs (s, a). a is the predicted action for state s from DRL model.

SDTs are fitted with transitions in the training set using supervised learning. During the execution phase, SDT generates a decision path for input state s. The feature weights in SDT and the decision path are then utilized in visualization module to support the behavior explanations.

B. Visualization Module

The distillation module is not sufficient to help users understand the agent behaviors because there is too much redundant information in the SDT model. Therefore, we implement a visualization module that provide explanations efficiently. The visualization module provides users the explanation information extracted from the distillation module with a graphical interface. Specifically, the visualization module contains (1) a tree plot showing feature weights of each node and decision path in a tree-structured image and (2) a trajectory plot showing all aircraft flying along their routes in the structured airspace with precise explanations.

1) Tree Plot: Each non-leaf node in SDT processes all input features with a one-layer network, so the feature weights give the information on how features influence the decision in one node. The features weights and output values of nodes along the decision path can provide the behavior explanations on how the hierarchical decision is made.

The tree plot illustrates feature weights of all SDT nodes in a tree structure. Feature weight of each node is visualized as a heatmap. Decision path is demonstrated with a dense black arrow connecting nodes along the decision path.

The explanation information is projected into the tree plot following these rules:

- Each feature weight is represented as a colored square in the heatmap.
- Feature weights of the same aircraft is drawn in the same row of heatmap.
- The higher the absolute value of weight is, the larger the size and the deeper the color is for the square. This also implies that the feature is more influential on the decision in the current layer.
- A red square means the increase in the feature value will increase the node output and a blue one means the increase in value will decrease the node output.

2) Trajectory Plot: While the tree plot gives a comprehensive explanation of behaviors, the trajectory plot only shows the most influential features for decision-making with both visual symbols and text. The simple structure of trajectory plot provides users with the most important information to understand the agent behaviors.

There are three main components in a trajectory plot: (1) all aircraft flying along routes, (2) highlighted influential factors, and (3) text boxes showing action information and ownship behavior explanations. Following rules are applied to demonstrate vital information with the trajectory plot:

- For each node in decision path, the feature whose weight has the largest absolute value and the aircraft which this feature belongs to will be selected as important feature and important aircraft. For an SDT with depth n, there will be n important features and n important aircraft for the trajectory plot. Their icons will be emphasised in the trajectory plot.
- Different symbols are used to emphasize features. For example, size of important aircraft will be increased. Distance feature will be drawn as an orange solid line.

C. Integration of Two Modules

To provide explanations of agent behaviors for aircraft separation assurance, we integrate the distillation module and visualization module together. We illustrate the architecture of the integrated framework in Figure 1.

At each time step, one forward pass for input state s is executed in distillation module. The decision path p is generated and transited to the visualization module. Based on the feature weights and decision path, visualization module draws tree plot and trajectory plot to provide users with the explanations of agent behaviors.

D. Efficiently Implementing SDT

To further improve model performance, we discuss techniques that boost the performance of SDT: use of batch normalization, removal of regularizations and label balancing.

a) Batch Normalization: Batch normalisation (BN) [45] is a technique to normalize layer input in neural networks. BN operator subtracts the mean value of mini-batch and subsequently divides the centered input by the standard deviation of mini-batch during training. In our problem setting, feature scales vary greatly and BN helps normalize the input and provide interpretations. We implement a BN layer prior to the root of SDT, which means the input of each node has been normalized.

b) Removal of unnecessary regularizations: In original SDT, β is an inverse temperature parameter to avoid very soft decisions and L1 regularization is used to avoid overfitting. We found that β and L1 regularization not only harm the performance of SDT but also decrease the training speed. The reason may be that SDTs in our setting can learn deterministic decisions without β and the sparse parameters brought by L1 regularization harms the performance of shallow tree model. So we remove both of them.

c) Label Balancing: The action distribution of transitions in training set is highly skewed because the aircraft maintain the speed in most situations and speed changes are discouraged by the reward function. We enlarge the training set and balancing its action distribution to provide enough samples of speed changes for model training.

VI. Numerical Experiments

A. Simulator

We utilize the BlueSky air traffic simulator [44] built in Python to evaluate the performance of our proposed SESAME framework. We can easily obtain the state information of all aircraft with the BlueSky simulator.

B. Case Studies

In this work, we consider three case studies *A, B*, and *C* with different routes and intersections as shown in Figure 2. The termination of simulations is when all aircraft in the airspace have reached their terminal states. Our goal is to distill DRL models into SDTs and provide stepwise explanations of aircraft behaviors in real time with the SESAME framework. These are very challenging cases to explain since DRL agents must learn a strategy and the same behavior could result from various reasons based on different situations. At the same time, DRL model will provide speed advisories to maintain separation assurance and focus on the goal that every aircraft exits the sector without conflict.

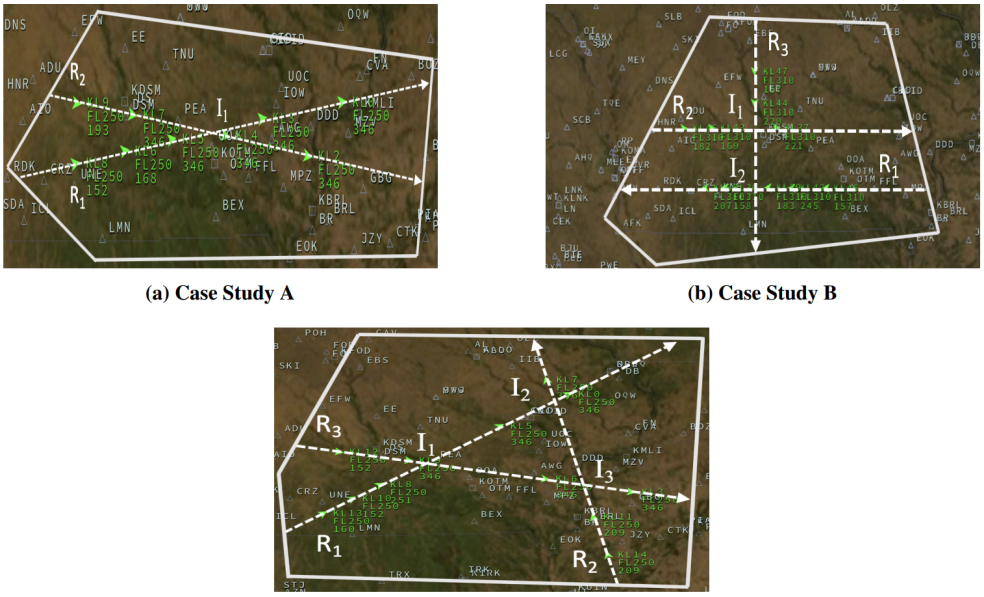
In our settings, the same SESAME framework and DRL model are implemented on each aircraft. Each aircraft selects its own desired speed with the DRL model. The implemented SESAME framework provides explanations for the selected action. This decentralized execution setting increases the problem complexity since the explanation framework need to consider how the cooperation among aircraft influences the agent behaviors.

C. The DRL Model

In this work, SDTs are distilled from two state-of-the-art DRL models for aircraft separation assurance in structured airspace: D2MAV-NC [7], D2MAV-A [9]. The effectiveness of our frameworks is evaluated on both DRL models to validate its generalization. This setting increases the difficulty of the problem since SESAME framework needs to generalize well in different DRL models to provide explanations.

D2MAV-NC utilizes an actor-critic model that incorporates the PPO loss function to provide speed advisories for separation assurance with a fixed number of intruders. We follow the default settings in [7].

D2MAV-A integrates an actor-critic model and an attention network to maintain safety separation among a variable number of intruders. We follow the default settings in [9]. To make it comparable to D2MAV-NC, we fix the number of intruders in D2MAV-A to 7 because the number of intruders is smaller than or equal to 7 in most samples.



(a) Case Study A

(b) Case Study B

(c) Case Study C

Fig. 2: Case Studies for evaluation in the BlueSky simulation environment.

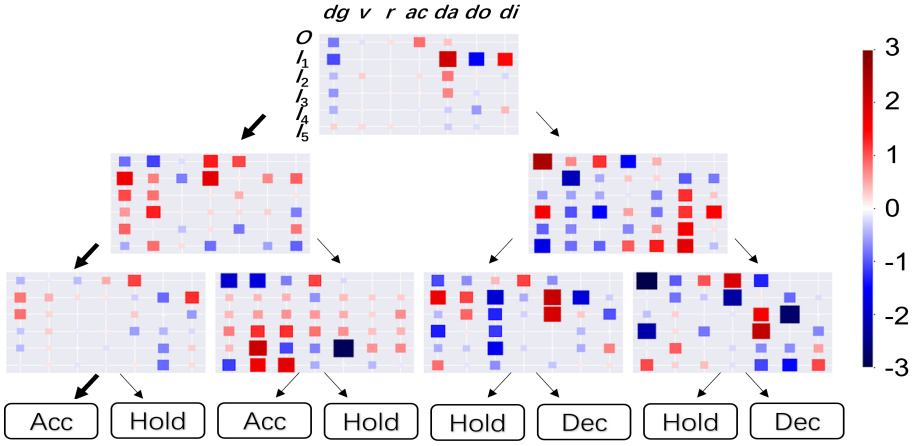


Fig. 3: Tree plot for SESAME framework using SDT-NC-BN with depth 3 for case A in one step. All non-leaf nodes process the same input state but assign different weights to features. Feature weight of each node is visualized as a heatmap. Each row in heatmap represents all features of one aircraft. The first row is for the ownship O. The other rows are for the five intruders I_1,...,I_5 sorted by distance to the ownship in ascending order. Each column represents value of the same feature for all aircraft. The features are distance to goal dg, current speed v, route identifier r, current acceleration ac, distance between ownship and the intruder da, distance between ownship and the intersection do, and distance between intruder and intersection di. Each colored square represents a feature weight. The higher the absolute value of weight is, the larger the size and the deeper the color is for the square. Red represents a positive influence and blue represents a negative influence. The dense black arrows represent the direction of decision path. Each leaf node gives a predicted action. *Acc* stands for acceleration. *Hold* stands for maintaining the current speed. *Dec* stands for deceleration.

D. Evaluation of model fidelity

We consider the SDT in our distillation module as a function approximator for the DRL model. We evaluate whether the predictions of distilled SDT model match those of the original DRL model. Specifically, we measure it with *fidelity* metric, which is defined as:

$$P(a_{\text{SESAME}} = a_{\text{DRL}}),$$

where a is output action based on model prediction. The subscripts stand for different models. Since fidelity measures the extent to which the decisions in SDT corresponds to those in the original DRL model, a higher fidelity score means that the behaviors of SDT and DRL models match better.

We report the fidelity scores of distilled SDT models with different depths for three case studies in Table I, Table II, and Table III. Each row shows results of a tree model with different tree depths. Each column shows results of 6 different models with the same depth. For evaluation, new transitions of 100 episodes are generated with DRL models. We compare the predicted actions from SDTs in these states with output actions from DRL models. The random policy baseline is 33.33%. To provide more comprehensive comparisons, we also report fidelity scores of HDTs as baseline. HDTs and SDTs are trained with the same training set. SDTs trained with D2MAV-NC and D2MAV-A are named as SDT-NC and SDT-A. SDTs with a BN layer are named as SDT-NC-BN and SDT-A-BN. HDTs trained with D2MAV-NC and D2MAV-A are named as HDT-NC and HDT-A.

TABLE I: Fidelity Scores(%) in Case A

Model	**Depth**					
	1	2	3	4	5	6
SDT-A	54.22	77.39	79.60	80.85	81.05	81.12
SDT-A-BN	55.84	79.44	82.64	84.48	85.76	86.64
SDT-NC	60.73	88.21	90.05	92.03	91.81	92.17
SDT-NC-BN	61.11	89.29	92.62	93.77	94.83	95.49
HDT-A	51.01	67.95	73.12	76.58	78.07	79.62
HDT-NC	57.83	73.93	76.08	82.35	82.95	84.42

TABLE II: Fidelity Scores(%) in Case B

Model	**Depth**					
	1	2	3	4	5	6
SDT-A	76.57	85.77	86.26	85.71	85.82	84.36
SDT-A-BN	77.53	87.97	90.42	90.92	92.22	92.21
SDT-NC	85.40	87.75	90.30	91.45	90.08	92.23
SDT-NC-BN	85.72	87.56	93.17	94.66	95.93	96.61
HDT-A	74.53	79.11	85.15	88.78	89.36	90.49
HDT-NC	74.90	78.74	84.30	85.32	87.33	89.49

TABLE III: Fidelity Scores(%) in Case C

Model	**Depth**					
	1	2	3	4	5	6
SDT-A	52.63	70.72	76.09	78.01	79.23	81.85
SDT-A-BN	54.24	71.91	77.09	80.30	84.21	85.90
SDT-NC	63.70	92.72	95.20	95.78	96.56	96.61
SDT-NC-BN	63.51	92.94	96.09	97.03	97.08	97.16
HDT-A	43.53	56.68	62.19	65.49	69.29	74.44
HDT-NC	65.83	86.42	88.71	91.36	92.00	92.35

Based on results in the same column, we find that the proposed SESAME framework has higher fidelity scores than baseline HDTs or random policy given the same depth in almost all cases except for case B with depth 4, 5, 6. This shows that our proposed framework can match the behaviors of original DRL models better than HDTs and random policy. Since the network in each leaf needs to generalize for a huge number of different input states, it is not unexpected that the tree policy may not cover the DRL model perfectly. The results also show that our framework generalizes well with different DRL models and BN helps improve model fidelity.

Based on results in the same row, we notice that a tree with more layers tend to perform better in terms of model fidelity given the same true structure, which implies that a deeper tree can match the DRL model better. This should relate to the fact that deeper trees have the capacity to divide the input space into more fine-grained cases and handle them better. However, the deeper the tree is, the more complex the tree policy is and the harder it is to explain. So this also shows trade-off between model fidelity and interpretability.

While SDTs achieve high fidelity scores in this work, we notice that the outputs of SDTs in some transitions are different from those of the DRL model. In these cases, our SESAME framework will notify the users. Additionally, future work will explore combining explainable tree models with other XDRL methods such as saliency maps to address the disagreement between SDTs and DRL models.

E. Explanations with Tree Plot

SDTs rely on hierarchical decisions along the decision path. Non-leaf nodes focus on different input features. The weights of non-leaf nodes and the decision path can provide the explanations on which features influence the agent behaviors. In this subsection, we demonstrate how tree plot can provide behavior explanations. A tree plot for model SDT-NC-BN with depth 3 on case A is drawn in Figure 3.

In the heatmap of root node, we see that the square at cell (2,5) has the largest size in deep red. Since that square represents feature of distance between ownship and the closest intruder, the size and color show that root puts the highest positive weight on this feature. This means root node focuses on the positive influence of the distance between ownship and the closest intruder. The distance to intruders is vital because it is more likely that ownship may collide with an intruder when they are at a small distance. Compared with distant intruders, potential collision with the closest one is the most urgent. As value of this feature increases, it is more likely to traverse to node 2, which is its left child node.

Checking the nodes along the decision path, we find that node 2 focuses on the the acceleration information of the closest intruder at cell (2,4). The right child node namely node 3 focuses on the positive influence by the distance to goal of the ownship. The different focuses between node 2 and node 3 shows that the child nodes concentrate on different features based on the output of its parent node.

Explaining how non-leaf nodes in the bottom layer lead to different actions is vital. This can be achieved by examining the heatmaps of nodes in the bottom non-leaf layer. Node 4 on the decision path highlights the positive influence by distance from the closest intruder to intersection. So it is more

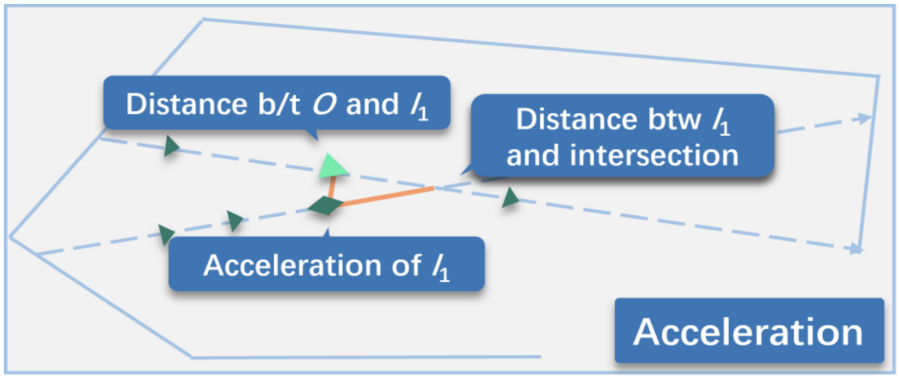


Fig. 4: Trajectory plot for SESAME framework using SDT-NC-BN with depth 3 in case A. All aircraft are drawn as triangles by default. Ownship is in light green and intruders are in deep green. In this case the important features are (1) The distance between ownship O and the closest intruder I_1, (2) the acceleration of the closest intruder, and (3) distance between the closest intruder and the intersection. The distance features are highlighted as orange solid lines and the accelerating aircraft is drawn as a diamond. The explanations and speed advisory are provided in text as well.

likely that acceleration action will be selected if the distance from the closest intruder to intersection becomes larger. This makes sense because ownship should accelerate to pass the intersection and avoid the collision when the closest intruder is still far from the intersection.

After we make explanations for nodes along the decision path, we give a summary explanation of the entire tree. At the root node, the left path rules out the deceleration action and the right path excludes the acceleration action. Each node in bottom layer can only select whether to maintain the speed or not. No bottom non-leaf node has access to both deceleration and acceleration actions.

F. Explanations with Trajectory Plot

In this subsection, we show how the trajectory plot provides precise behavior explanations with only important information. A trajectory plot for DRL model SDT-NC-BN with depth 3 on case A is drawn in Figure 4. Figure 4 and Figure 3 show the explanations for the same transition.

The ownship near the intersection accelerates in this example. The important features are identified as the features with the highest absolute value in each node along the decision path. The important features in this transition are (1) distance between ownship and the closest intruder, (2) acceleration of the closest intruder, and (3) distance between the closest intruder and intersection. These features can be extracted from the model weights of SDT. We also have introduced the importance of these features in the previous subsection. The distance features are highlighted with an orange solid line and the acceleration feature is emphasized as a diamond in the plot. Text explanations are also listed in separate text boxes on the trajectory plot. A text box in the low right corner shows the speed advisory.

Compared to the tree plot which provides a comprehensive and detailed explanation, trajectory plot illustrates a precise and efficient explanation with only important information. Integrating two plots together, the SESAME framework provides behavior explanations with both important information in trajectory plot and supplemental details in tree plot.

G. Evaluation of SDT Controller

TABLE IV: Performance of SDT in case A

Model	**Depth**					
	1	2	3	4	5	6
SDT-A	17.78	26.20	23.52	24.38	25.94	26.38
SDT-A-BN	19.34	26.82	26.04	29.36	28.67	29.07
SDT-NC	23.84	21.12	23.71	25.66	26.37	26.46
SDT-NC-BN	25.60	29.00	29.02	29.32	29.68	29.64
HDT-A	23.52	25.68	17.08	15.24	23.92	19.95
HDT-NC	22.06	23.16	22.72	23.10	21.90	22.30

TABLE V: Performance of SDT in case B

Model	**Depth**					
	1	2	3	4	5	6
SDT-A	25.52	25.46	25.52	25.34	25.26	25.56
SDT-A-BN	25.94	25.48	24.96	25.58	25.98	26.64
SDT-NC	27.06	27.18	26.68	27.32	27.72	26.12
SDT-NC-BN	28.12	25.84	29.56	28.78	29.80	29.90
HDT-A	25.36	25.02	24.94	24.27	23.57	25.18
HDT-NC	25.18	26.32	27.14	26.78	27.24	26.73

TABLE VI: Performance of SDT in case C

Model	**Depth**					
	1	2	3	4	5	6
SDT-A	17.19	20.45	18.66	19.94	19.31	23.33
SDT-A-BN	22.45	19.76	22.27	21.85	24.64	24.28
SDT-NC	19.70	22.46	26.34	26.38	26.26	27.76
SDT-NC-BN	25.64	21.83	26.28	27.70	28.50	28.44
HDT-A	18.70	13.09	13.68	15.72	13.98	14.33
HDT-NC	20.14	20.74	18.54	18.10	20.66	19.30

Since the behaviors of distilled SDTs match those of the DRL models well, we evaluate whether the distilled SDTs can directly control the agent for the separation assurance task instead of only providing explanations. Specifically, we use the distilled SDTs as controllers for the aircraft separation assurance task. We define the performance score as the number of aircraft exiting the sector without conflict among all 30 aircraft. We report the average performance of 50 episodes for three cases in Table IV, Table V, and Table VI. The HDT controllers are used to provide a fair comparison as baseline models in this experiment. The performances of random policy baseline are 12.20, 19.76, and 12.05 in case A, B, and C. The DRL models can achieve an optimal performance with no collision in these three cases.

Results show that SDT controllers work better than random policy and HDTs in all cases. The reason may be that SDT can learn a hierarchical structure to make decisions relying on all features at the same time. SDTs work better in case A and B than in case C because case C has a more complex structure with three routes and three intersections.

We notice that in some cases (e.g., SDT-NC-BN with depth 2 in case A, SDT-NC-BN with depth 3 in case B) SDT can gain near-optimal safety performance with only 2 or 3 layers. This implies the potential for SDTs to both make decision and provide explanations in safety-critical systems.

VII. Conclusions

The behavior explanation of DRL models for aircraft separation assurance tasks with complex and high-dimensional state space is an extremely difficult problem. In this paper, we propose an explanation framework to distill the complex DRL models into SDTs and visualize the information extracted from the model weights along the decision path to provide behavior explanations. This framework can provide visual explanations of agent behaviors for aircraft separation assurance task in a structured en route airspace sector.

We demonstrate the effectiveness of our framework on separation assurance problem with extensive experiments. We show that the behaviors of distilled SDTs match those of original DRL models well. We also show that the proposed framework can provide behavior explanations by visualizing learned features. Furthermore, we demonstrate the generalization of our framework by distilling SDT policies from two different state-of-the-art DRL-based separation assurance models. Finally, we show that the distilled SDTs can be used directly as the controllers to perform safety separation assurance with a good performance.

In summary, our proposed framework can explain the behaviors of DRL-based agent for aircraft separation assurance. This framework will be useful in real-world safety-critical systems where the model explanation is of high importance. The promising results encourage us to further explore the effectiveness of the framework and its extensions for other safety-critical applications in the future.

Acknowledgment

This project is supported by the NASA Grant 80NSSC21M0087 under the NASA System-Wide Safety (SWS) program.

References

[1] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[2] Y. Pan, C.-A. Cheng, K. Saigol, K. Lee, X. Yan, E. Theodorou, and B. Boots, "Agile autonomous driving using end-to-end deep imitation learning," *arXiv preprint arXiv:1709.07174*, 2017.

[3] J. Mollinga and H. van Hoof, "An autonomous free airspace en-route controller using deep reinforcement learning techniques," *CoRR*, vol. abs/2007.01599, 2020. [Online]. Available: https://arxiv.org/abs/2007.01599

[4] P. N. Tran, D.-T. Pham, S. K. Goh, S. Alam, and V. Duong, "An interactive conflict solver for learning air traffic conflict resolutions," *Journal of Aerospace Information Systems*, vol. 17, no. 6, pp. 271–277, 2020.

[5] S. Ghosh, S. Laguna, S. H. Lim, L. Wynter, and H. Poonawala, "A deep ensemble method for multi-agent reinforcement learning: A case study on air traffic control," in *Proceedings of the Thirty-First International Conference on Automated Planning and Scheduling, ICAPS 2021, Guangzhou, China (virtual), August 2-13, 2021*. AAAI Press, 2021, pp. 468–476. [Online]. Available: https://ojs.aaai.org/index.php/ICAPS/article/view/15993

[6] W. Guo, M. Brittain, and P. Wei, "Safety enhancement for deep reinforcement learning in autonomous separation assurance," in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, 2021, pp. 348–354.

[7] M. Brittain and P. Wei, "Autonomous separation assurance in an high-density en route sector: A deep multi-agent reinforcement learning approach," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 3256–3262.

[8] M. W. Brittain and P. Wei, "One to any: Distributed conflict resolution with deep multi-agent reinforcement learning and long short-term memory," in *AIAA Scitech 2021 Forum*, 2021, p. 1952.

[9] M. Brittain, X. Yang, and P. Wei, "A deep multi-agent reinforcement learning approach to autonomous separation assurance," *CoRR*, vol. abs/2003.08353, 2020. [Online]. Available: https://arxiv.org/abs/2003.08353

[10] R. Jonschkowski and O. Brock, "Learning state representations with robotic priors," *Autonomous Robots*, vol. 39, no. 3, pp. 407–428, 2015.

[11] D. Jarrett, A. Hüyük, and M. Van Der Schaar, "Inverse decision modeling: Learning interpretable representations of behavior," in *International Conference on Machine Learning*. PMLR, 2021, pp. 4755–4771.

[12] N. Frosst and G. Hinton, "Distilling a neural network into a soft decision tree," *arXiv preprint arXiv:1711.09784*, 2017.

[13] J. R. Quinlan, "Induction of decision trees," *Machine learning*, vol. 1, no. 1, pp. 81–106, 1986.

[14] Z. Ding, P. Hernandez-Leal, G. W. Ding, C. Li, and R. Huang, "Cdt: Cascading decision trees for explainable reinforcement learning," *arXiv preprint arXiv:2011.07553*, 2020.

[15] Y. Coppens, K. Efthymiadis, T. Lenaerts, A. Nowé, T. Miller, R. Weber, and D. Magazzeni, "Distilling deep reinforcement learning policies in soft decision trees," in *Proceedings of the IJCAI 2019 workshop on explainable artificial intelligence*, 2019, pp. 1–6.

[16] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[17] R. Dalmau and E. Allard, "Air traffic control using message passing neural networks and multi-agent reinforcement learning."

[18] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," in *NIPS Deep Learning Workshop*, 2013.

[19] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 30, no. 1, 2016.

[20] M. Ribeiro, J. Ellerbroek, and J. Hoekstra, "Determining optimal conflict avoidance manoeuvres at high densities with reinforcement learning," in *Proceedings of the Tenth SESAR Innovation Days, Virtual Conference*, 2020, pp. 7–10.

[21] R. Isufaj, D. Aranega Sebastia, and M. A. Piera, "Towards conflict resolution with deep multi-agent reinforcement learning," in *Proceedings of the 14th USA/Europe Air Traffic Management Research and Development Seminar (ATM2021), New Orleans, LA, USA*, 2021, pp. 20–24.

[22] Z. Wang, H. Li, J. Wang, and F. Shen, "Deep reinforcement learning based conflict detection and resolution in air traffic control," *IET Intelligent Transport Systems*, vol. 13, no. 6, pp. 1041–1047, 2019.

[23] B. Wulfe, "Uav collision avoidance policy optimization with deep reinforcement learning," 2017.

[24] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[25] D.-T. Pham, N. P. Tran, S. K. Goh, S. Alam, and V. Duong, "Reinforcement learning for two-aircraft conflict resolution in the presence of uncertainty," in *2019 IEEE-RIVF International Conference on Computing and Communication Technologies (RIVF)*. IEEE, 2019, pp. 1–6.

[26] N. P. Tran, D.-T. Pham, S. K. Goh, S. Alam, and V. Duong, "An intelligent interactive conflict solver incorporating air traffic controllers' preferences using reinforcement learning," in *2019 Integrated Communications, Navigation and Surveillance Conference (ICNS)*. IEEE, 2019, pp. 1–8.

[27] H. Wen, H. Li, Z. Wang, X. Hou, and K. He, "Application of ddpg-based collision avoidance algorithm in air traffic control," in *2019 12th International Symposium on Computational Intelligence and Design (ISCID)*, vol. 1. IEEE, 2019, pp. 130–133.

[28] D.-T. Pham, N. P. Tran, S. Alam, V. Duong, and D. Delahaye, "A machine learning approach for conflict resolution in dense traffic scenarios with uncertainties," in *ATM Seminar 2019, 13th USA/Europe ATM R&D Seminar*, 2019.

[29] A. Verma, V. Murali, R. Singh, P. Kohli, and S. Chaudhuri, "Programmatically interpretable reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2018, pp. 5045–5054.

[30] J. Kim, S. Moon, A. Rohrbach, T. Darrell, and J. Canny, "Advisable learning for self-driving vehicles by internalizing observation-to-action rules," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9661–9670.

[31] G. Cideron, M. Seurin, F. Strub, and O. Pietquin, "Self-educated language agent with hindsight experience replay for instruction following," 2019.

[32] S. Greydanus, A. Koul, J. Dodge, and A. Fern, "Visualizing and understanding atari agents," in *International Conference on Machine Learning*. PMLR, 2018, pp. 1792–1801.

[33] N. Puri, S. Verma, P. Gupta, D. Kayastha, S. Deshmukh, B. Krishnamurthy, and S. Singh, "Explain your move: Understanding agent actions using specific and relevant feature attribution," in *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. [Online]. Available: https://openreview.net/forum?id=SJgzLkBKPB

[34] X. Chen, Z. Wang, Y. Fan, B. Jin, P. Mardziel, C. Joe-Wong, and A. Datta, "Reconstructing actions to explain deep reinforcement learning," *arXiv preprint arXiv:2009.08507*, 2020.

[35] Z. Juozapaitis, A. Koul, A. Fern, M. Erwig, and F. Doshi-Velez, "Explainable reinforcement learning via reward decomposition," in *IJCAI/ECAI Workshop on explainable artificial intelligence*, 2019.

[36] Z. Lin, K.-H. Lam, and A. Fern, "Contrastive explanations for reinforcement learning via embedded self predictions," in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=Ud3DSz72nYR

[37] B. Lütjens, M. Everett, and J. P. How, "Safe reinforcement learning with model uncertainty estimates," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8662–8668.

[38] M. Matarese, S. Rossi, A. Sciutti, and F. Rea, "Towards transparency of td-rl robotic systems with a human teacher," *arXiv preprint arXiv:2005.05926*, 2020.

[39] S. Karakovskiy and J. Togelius, "The mario ai benchmark and competitions," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 4, no. 1, pp. 55–67, 2012.

[40] G. Liu, O. Schulte, W. Zhu, and Q. Li, "Toward interpretable deep reinforcement learning with linear model u-trees," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2018, pp. 414–429.

[41] A. Silva, M. Gombolay, T. Killian, I. Jimenez, and S.-H. Son, "Optimization methods for interpretable differentiable decision trees applied to reinforcement learning," in *International conference on artificial intelligence and statistics*. PMLR, 2020, pp. 1855–1865.

[42] N. Dahlin, K. C. Kalagarla, N. Naik, R. Jain, and P. Nuzzo, "Designing interpretable approximations to deep reinforcement learning with soft decision trees," *arXiv preprint arXiv:2010.14785*, 2020.

[43] R. N. Haksar and M. Schwager, "Distributed deep reinforcement learning for fighting forest fires with a network of aerial robots," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1067–1074.

[44] J. M. Hoekstra and J. Ellerbroek, "Bluesky ATC simulator project: an open data and open source approach," in *Proceedings of the 7th International Conference on Research in Air Transportation (ICRAT)*, vol. 131. FAA/Eurocontrol USA/Europe, 2016, p. 132.

[45] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*. PMLR, 2015, pp. 448–456.