Visual robot navigation incorporating causal models in deep reinforcement learning

Nilda G. Xolo-Tlapanco[1,2], Eduardo F. Morales[1], L. Enrique Sucar[1], and J. Ernesto Gomez-Balderas[2]

[1] Instituto Nacional de Astrofisica, Optica y Electrónica, Santa María Tonazintla, Puebla, México (nilda.xolo,emorales,esucar)@inaoep.mx
[2] GIPSA-Lab, Univ. Grenoble Alpes, CNRS, Grenoble INP, 38000 Grenoble, France
gomezbaj@univ-grenoble-alpes.fr

Abstract. Deep reinforcement learning can be used for visual navigation in mobile robots but needs substantial computational resources and long training time. To reduce exploration time and improve adaptation to novel situations or environments, we created an algorithm to learn a Causal Bayesian Network of the task and use it in a Deep Reinforcement Learning algorithm (DQN). Experimentally, we show its effectiveness in simulated environments for a visual robot navigation task.

Keywords: Causal Reinforcement Learning · Deep Reinforcement Learning · Micro Air Vehicles.

1 Introduction

The use of Unmanned Aerial Vehicles (UAV) or drones, has recently increased, in recent years, due to the growth interest in using them for entertainment, but also for military, agriculture ([22]), delivery services ([9]), and rescue applications ([18]), among others. Their principal disadvantage is the need for a human pilot to control it, so multiple research works have used different artificial intelligence techniques to develop autonomous pilots ([12][8] [11]). One of the most popular techniques is Reinforcement Learning (RL) which has been used for collision avoidance and seeking an objective/goal in different environments ([4] [25] [14]). RL algorithms require a large amount of data and long times for training. A promising alternative is to use causal discovery to construct a causal model of the environment which can help accelerate the training phase.

Causal reinforcement learning integrates causal inference into RL, with the aim of utilizing the underlying causal relationships in the environment. Incorporating knowledge from a causal model and their construction into RL algorithms, e.g., [7] [28] [19], can greatly speed up the learning process by reducing the need for extensive exploration, but their use in robotics task has not been yet proved. In this research, we show how it can be used to autonomously control navigation tasks in a drone in a simulated environment and build two algorithms: one that learns a Causal Bayesian Network (CBN) of the task and environment during training of the DQN algorithm; and another one (with two variants) that uses a CBN to guide the selection of actions in the DQN algorithm.

2 Fundamentals

Reinforcement learning studies sequential decision problems. Mathematically, we can formalize these problems as Markov Decision Processes (MDPs). An MDP can be formally described by a 4-truple (S,A,P,R) where:

- S is a state space in which the process of evolution takes place.
- A is a set of all possible actions that control the state dynamics.
- $P_a(s, s') = Pr(S_{t+1} = s'|S_t = s, A_t = a)$ denotes the probability of transition (at time t) from state s to state s' under action a.
- $R_a(s, s')$ provides the immediate reward after transition from state s to s' with action a.

MDPs allow us to model the state evolution dynamics of a stochastic system when this system is controlled by an agent choosing and applying the actions a_t at every time step t. The procedure of choosing such actions is called action policy or strategy and is written as π. A policy specifies which action to take at a particular state. Solving a Markov decision problem implies searching for a policy that optimizes a performance (or optimality) criterion. One way to solve MDPs, particularly when the state and action spaces are too large for tabular methods, is the use of Deep Reinforcement Learning (DRL).

DRL combines the perception capabilities of deep learning with the decision-making capabilities of reinforcement learning by using deep neural networks, this can handle more complex scenarios with less engineered feature extraction. One of the most commonly used DRL algorithms is Deep Q-Learning (DQL), which updates the Q-values using the Bellman equation:

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left[r + \gamma \max_{a'} Q(s', a') - Q(s, a) \right]$$

Where:

- $Q(s, a)$ is the state-action value function, representing the expected utility of taking action a in state s.
- α the learning rate ($0 < \alpha \leq 1$) determines to what extent newly acquired information overrides old information.
- r is the reward received after taking action a.
- $\max_{a'} Q(s', a')$ is the maximum predicted reward obtainable from the next state s', over all possible actions a'.
- γ is the discount factor ($0 \leq \gamma < 1$). It represents the difference in importance between future rewards and immediate rewards.

Traditional Q-Learning stores Q-values in a Q-table, but this is infeasible for large, continuous state spaces. DQL solves this by using a neural network to approximate Q-values, inputting states and outputting actions but it needs extensive interactions to gather enough data for training. This data can be used for action selection or causal discovery.

2.1 Causal Discovery

Causal Modeling attempts to solve questions about possible causes by providing explanations of phenomena (effects) as the result of previous phenomena (causes). Several writers [17][15] specify three conditions that must be met in order to infer the existence of a causal relationship between two variables, X and Y: (1) a covariation between X and Y, (2) a time-ordered asymmetry, and (3) the elimination of other possible causes. Graphs are used to represent dependencies, where connected variables are dependent, and independent relations are implicit. Causal Bayesian Networks (CBNs) offer a framework for causal inference and prediction and represent stronger assumptions than Bayesian networks; CBNs assume all relationships correspond to actual causal connections [26].

The combination of RL and causal modeling is a relatively new field called Causal Reinforcement Learning (CRL). CRL is a suite of algorithms incorporating additional assumptions or prior causal knowledge into RL to analyze and understand the causal mechanisms underlying actions and their consequences. This enables agents to make more informed and effective decisions for more effective model learning, policy evaluation, or policy optimization [5]. It is generally divided into two categories, the first category is based on the prior causal information, where such methods typically assume the causal structure about the environment is given *a priori* by experts; while in the second category the causal information has to be learned [27]. Our work belongs to the second category, we developed an algorithm to learn the causal information from the interactions of the RL agent with the environment to later use it in policy construction.

3 Related Work

3.1 Deep reinforcement learning

DRL has been widely used to develop autonomous drone pilots to avoid obstacles and achieve their goals. Darwish et al. [4] introduced a model-free DRL method using a depth-RGB camera, outperforming DQN in intercepting specified targets. Shin et al. [25] demonstrated a drone navigating 3D obstacles with a dual input of an RGB camera and depth map for precise path finding. Cetin et al. [2] proposed DRL for autonomous navigation in suburban environments using depth images. Kersand et al. [14] used depth images and heading to train DQL algorithms. A challenge in training DRL agents is the time and data required. Causal models can be used for interpretability, task transfer, and faster training.

3.2 Causal Modeling

Causal Modeling (CM) needs rigorous data collection and expert knowledge. In machine learning, it has been used mainly to generate explanations. Shi et al. [24] introduce a model for interpreting causal relationships in a temporal-spatial context, capturing the causal connections between consecutive observations and decisions made by an RL agent. In [13], the authors use structural causal models

to make human insight explicit in the causal relations used during the development of AI systems. Cetin et al. [3] use causal information to explain why the agent performs an specific action. It is based on the generation of a saliency map to identify the critical regions in an input image that influences the predictions made by the DQN agent. Diehl et al. [6] learn a CBN from simulation data to learn a cause-effect model of the environment, generating causal explanations.

3.3 Causal Reinforcement Learning

As we mentioned before, CRL is divided into two categories. The first category is based on prior causal information. Feliciano et al. [7] demonstrate better performance even with partial and spurious relationships by integrating a CM into Q-Learning in the light switch control tasks. Zhu et al. [28] use DQL in the Emotional Pendulum and Windy Pendulum tasks. Gonzalez et al.[10] introduced a decision-making approach for agents operating in environments characterized by uncertainty and underlying causal dynamics. This approach enabled the agent to continually update beliefs about the causal environment based on interactive outcomes. In their experimental setup, it was presupposed that the agent had prior knowledge of the causal framework governing the environment.

In the second category, the causal information has to be learned. Méndez et al. [19] learn a Causal Dynamic Bayesian Network for each of the agent actions and uses those models to improve the action selection process in the Coffee task. The same authors [20] developed a framework for simultaneously learning and using causal models to speed up policy learning in online MDPs, evaluated in the Coffee, Taxi and Taxi Atari tasks. MOCODA [21] applies a learned factored dynamics model to an augmented distribution of states and actions to generate counterfactual transitions for RL, and is used to train an offline RL agent to solve a robotics manipulation task.

It is clear that causal knowledge can help to accelerate reinforcement learning; however, there are very few approaches that learn causal models from data generated in RL and it has not been used for autonomus drones navigation.

4 Methodology

Data recollected by the agent during RL is used to learn a Causal Bayesian Network, which in turn is used in the action selection process of RL. We select Deep Q-Learning (DQN) as the algorithm to implement the visual navigation task in the drone. DQN uses two strategies to facilitate a more stable learning process: (i) An experience replay buffer, which stores the experiences of the agent and from which random samples are selected for the gradient descent process, (ii) the use of two networks to reduce the variance of the gradients, one fixed and used as a reference, and other one which is updated. The algorithm is described in Algorithm 1, the modifications on DQN are lines 5-7, where the Causal Bayesian Network is learned, and lines 8 and 14, where the Causal Bayesian Network is used.

Algorithm 1 Deep Q-Learning Algorithm

```
1: Initialize replay memory D to capacity N, action-value function Q with random
   weights θ and target action-value function Q̂ with weights θ⁻ = θ
2: for episode = 1, M do
3:     Initialize sequence s_1 = {x_1} and preprocessed sequence φ_1 = φ(s_1)
4:     for t = 1, T do
5:         if step % k == 0 then
6:             Learn the Causal Bayesian Network
7:         end if
8:         Set the probability of each action to reach a negative and positive reward
   with the evidence of the state t
9:         rand = random number
10:        if rand < explorationRate then
11:            action a_t = random action
12:        else
13:            action a_t = MaxIndex(qValues)
14:            Use of the Causal Bayesian Network (Algorithm 3)
15:        end if
16:    end for
17: end for
```

4.1 Learning of the Causal Bayesian Network

The structure of the CBN is learned while training the agent in DRL using the Hill Climb search algorithm and the BIC score. Once we have the structure of the directed graph, we need to learn the parameters. For this, we calculate the Markov blanket of the reward node and delete synchronous links in the graph. To perform the inference with the observable state, we use the variable elimination algorithm [16]. The learning of the CBN is described in Algorithm 2, where k is a predefined number of steps to update the Causal Bayesian Network, and *numActions* is the number of actions in the actions set.

4.2 Use of the CBN in DQN

If the CBN model is already known, it can be used from the onset of the learning process. However, when the CBN needs to be learned, the RL algorithm must first accumulate sufficient data, this is necessary to ensure that the data collected is adequate for learning an accurate approximation of the true CBN. Algorithm 3 describes how to use the CBN within DQN. At each step, it takes the state elements present in the Markov blanket of the reward node as *filtered evidence*, which is used in the inference of reaching each state value. We divide this inference into two arrays, for the probability of transition to a positive reward (*ProbAct*) and for the probability to transition to a negative reward (*ProbActNeg*). Lines 10-13 describes the normal behavior of DQN, after that, depending of the probability of consulting the CNB *ProbCausalModel* (lines 14-26), the algorithm decides which action to take. If the action selected has a high probability (*ThresholdProb*) to reach a negative reward, the action is eliminated as

Algorithm 2 Learning the Causal Bayesian Network

```
1: At each step in the QDN algorithm save the state representation at time t, t + 1,
   reward, and accumulated reward in the variable data
2: if step % k == 0 then
3:     for i = 0 to numActions do
4:         Check for the previous model
5:         if there is a previous model then
6:             Start with a pre-defined structure for the CBN
7:         else
8:             Start with a random initial structure for the CBN
9:         end if
10:        model[i]= HillClimbSearch(BIC score, data)
11:        Delete synchronous links, calculate the Markov blanket for the node Reward
   and create/update CPTs
12:        inference[i] = VariableElimination(model[i])
13:    end for
14: end if
```

an option and another one is randomly selected (lines 16-20). If the selected action has a high probability to obtain a positive reward, the action is executed (lines 22-24). We define a variant of the previously described method where the decision to consult the CBN is performed before the behavior of the standard DQN.

5 Experimental results

For the implementation of DQN for visual navigation in the drone, we followed the implementation of Anas et al. [1] and adapted it to our task.

State space: The agent's state spaces consist of: (i) An image with dimensions 84x84x3 pixels used to evaluate the performance of the baseline algorithm without causal models. (ii) An image with dimensions 84x84x3 pixels + 9 values for the learning and use of the CBN.

Action space: the agent can take eight discrete actions namely: *forward* and *backward* moves (move the drone in the y axis), *turn left* and *turn right* (change the orientation of the drone in the x axis), *right* and *left* (move the drone in the x axis), and *ascend* and *descend* (move the drone in the z axis).

Target: a goal image.

Reward: calculated from the following formula.

$$\text{reward} = \begin{cases} 100 - \text{Dist. to goal} - |\text{angle to goal}| & \text{if the target is recognized} \\ 1000 & \text{if the goal is reached} \\ -1000 & \text{if the drone crashes} \\ -9 & \text{otherwise} \end{cases}$$

Termination conditions: 1) The agent reaches the goal within 0.2 meters. 2) The agent collides with an obstacle. 3) The number of steps exceeds 200.

Algorithm 3 Use of the Causal Bayesian Network for DQN

```
At each step initialize Evidence of the state t
for i = 0 to numActions do
    filter evidence of the Markov blanket of the node reward
    result = inference[i] for variable reward with filtered evidence
    ProbActNeg[i]=Probability of the action to obtain a negative reward
    ProbAct[i]=Probability of the action to obtain a positive reward
end for
initialize ProbCausalModel and ThresholdProb
rand = random number
if rand < explorationRate then
    action a_t = random action
else
    action a_t = MaxIndex(qValues)
    rand = random number
    if rand < ProbCausalModel then
        if ProbActNeg[action a_t] > ThresholdProb then
            ActionNegative = action
            while action a_t == ActionNegative do
                action a_t = random action
            end while
        else
            if ProbAct[action a_t] < ThresholdProb then
                action a_t = MaxIndex(ProbAct)
            end if
        end if
    end if
end if
```

Network Architecture: The Prediction and Target Networks share the same structure.

- Input Layer: 84x84x3 RGB image.
- Convolutional Layers (ReLU activated):
  - Layer 1: 32 filters, 8x8 kernel, stride 4.
  - Layer 2: 64 filters, 4x4 kernel, stride 2.
  - Layer 3: 64 filters, 3x3 kernel, stride 1.
- Flatten Layer: Converts output to 1D.
- Dense Layer: 512 units for processing features.
- Output Layer: 8 nodes for Q-values of actions.
- Loss Function: Mean Squared Error (MSE).
- Optimizer: RMSprop, learning rate 0.00025, discount 0.9, epsilon 0.00000001.

To represent the information needed for the construction of the CBN we used 9 values: (i) Distance of five defined sections of the image (center, top left, top right, bottom left, and bottom right). In each section we select a number specific of random points (at least 20 points), taking the smallest distance as the overall distance of the drone to the objects in this section. (ii) A boolean value if the

goal is in the field of view. (iii) Distance and angle to the goal, and (iv) Altitude of the drone. To obtain these values we need three sensors: an RGB camera to process the image to detect the goal, a Depth Camera to obtain the distances, and a barometer or similar to obtain the altitude of the drone.

The goal is a sign of "Heliport". Usually, the goal is defined with a set of coordinates, but we try not to depend on the agent seeing the goal all the time, and, in real cases, access to the real coordinates requires the help of GPS or other sensors that are more difficult to access. To enable the drone to recognize the goal, we trained a custom YOLO v8 model [23] for 50 epochs and obtained a precision of 0.943 and a recall of 0.902 for goal detection, using a dataset of 1070 images from simulations and real-world data. The YOLO model returns a list of detected objects, each with a bounding box coordinates and a score, representing the confidence in the detected object. We set a threshold of 0.7 for the confidence score. Once the model detects the object and the bounding box coordinates are obtained, the distance is calculated in the same way as the sections. We also establish an angle between the center of the image and the center of the goal.

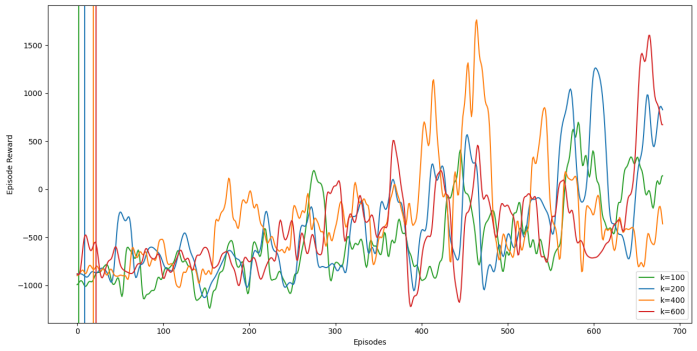
For experimental purposes we construct a CBN of the task to use in the reinforcement learning algorithm, taking into account the 9 values obtained from the RGB and depth image. To simplify and reduce the observation space we discretized all the values, taking two values for the distance: close and far; a boolean value for the goal in sight; three values for the angle: center, far left and far right, and three values for the altitude: good, close and far ground.

5.1 Results

We begin our experiments by using the state representation for learning and using the CBN to determine the optimal value for k. The parameter k dictates the point at which the CBN learning algorithm starts its learning process. We evaluated its effectiveness across various k values: 100, 200, 400, and 600. Their performance is illustrated in Figure 1, the vertical axis represents the reward obtained in each episode after applying a moving average of size 20 to smooth out short-term fluctuations and highlight longer-term trends, rewards range from below -1000 up to approximately 1500. The horizontal axis tracks the number of episodes, ranging from 0 to 700. Before the first k steps the algorithm behaves as DQN, after k steps the algorithm can consult the current CBN model.

Lower k values seem to have more stability compared to higher k values, which exhibits more dramatic ups and downs. This indicates that lower k values provide a more consistent learning experience or adaptation, whereas higher k values, while capable of achieving higher peaks, may introduce volatility. For this reason, we select the value of 200 for k to compare with the other algorithms.

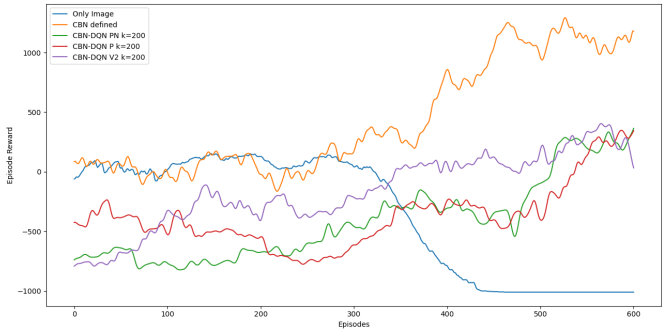
To evaluate the performance of our algorithm, we compared the DQN algorithm as baseline (only image) with the proposed algorithm that adds the discrete state representation, using: (i) the defined Causal Bayesian Network (ii) learning the CBN (CBN-DQN P) (iii) a variant of the algorithm considering the probability of receiving a positive and negative reward from the CBN to decide



,

Fig. 1. Episode reward obtained after 700 episodes in the learning and use of the CBN in the DQN algorithm with k equals to 100, 200, 400, and 600. The vertical lines indicate from which episode the CBN is learned and subsequently used and updated. We utilize a moving average of size 20.

which actions to take versus considering only the probability of positive rewards for the first version (CBN-DQN PN) and (iv) learning the CBN in the second version of the algorithm (CBN-DQN V2), all with k=200



,

Fig. 2. Episode reward obtained after 700 episodes with base DQN, DQN with the CBN defined, CBN-DQN PN, CBN-DQN P and CBN-DQN V2 with k=200, with a moving average of 50.

Their performance is illustrated in Figure 2, the vertical axis represents the reward obtained in each episode after applying a moving average of size 50. The best behavior is when the Causal Bayes Network is manually defined in advance because it is used from the beginning of the learning process and the

CBN is more accurate. Using only the image as state representation has not adequate information to reach the goal and the network learns to "do nothing". In comparison, the CBN-DQN V2 is the closest to the behavior of the algorithm with a predefined causal model, compared to use only the action with a higher probability of carrying a positive reward (CBN-DQN P) and using both (CBN-DQN PN). This could be because the first version of the algorithm benefits more from performing random actions at the first episodes depending of the exploration rate, so it might require more training episodes for better behavior compared to the version two of the algorithm where the probability of consulting the CBN is dominated by the exploration rate.

We also analyze the behavior of the CBN learning method. We can observe the evolution of the CBN for CBN-DQN V2, for the action forward in Figure 3. It begins with 4 connections between time t and time $t+1$ but at the end, it has 7 connections. It is interesting to note the dependence between the reward and the angle for the same $t+1$, which depends of seeing the goal and its angle goal in the previous time. This behavior of the learned CBNs helps us to note dependencies that we did not take into account in the manual definition of the network. Also, the lack of direct connections to the reward node can indicate that there was not sufficient information involving seeing the goal and consequently the angle and distance to the goal.

From the experiments we can conclude the following:

- A Causal Bayesian Network (given or learned) can be used to improve the performance of an RL agent in robotic tasks with a partially continuous space state.
- A Causal Bayesian Network can be learned from data obtained during the reinforcement learning process but more episodes are needed to obtain a good model.

6 Conclusions

In this work, we developed two algorithms: one that learns a Causal Bayesian Network of the task and environment during the training of the DQN algorithm

Fig. 3. Comparison of the first and last Causal Bayesian Network for action *forward*.

and another one (with two variants) that uses a CBN to guide the selection of the actions in the DQN algorithm. We tested the algorithms in the task of autonomous navigation of a drone in a simulated environment. The results show that the use of causal knowledge can accelerate learning a policy; and that it is possible to learn partial CBNs, while learning a policy, but more episodes are required to reach a stable and accurate CBN.

As future work, we need to train our algorithms for more episodes to prove the stability of the algorithm and confirm if the learned CBN reaches the accuracy of the model with the network previously defined. We also need to perform more tests on different environments to assess the generality of the proposed approach. We would also like to include an exploration strategy, in case the objective is out of sight. Finally, we believe that the learned CBNs can be transferred to other, although similar, domains where the causal relationships are still valid.
Acknowledgments The authors would like to acknowledge the funding support for this work from CDP-BOOT and CONAHCYT.

References

1. Anas, H., Ong, W.H., Malik, O.A.: Comparison of deep q-learning, q-learning and sarsa reinforced learning for robot local navigation. In: Kim, J., Englot, B., Park, H.W., Choi, H.L., Myung, H., Kim, J., Kim, J.H. (eds.) Robot Intelligence Technology and Applications 6. pp. 443–454. Springer, Cham (2022)
2. Çetin, E., Barrado, C., Muñoz, G., Macias, M., Pastor, E.: Drone navigation and avoidance of obstacles through deep reinforcement learning. In: 2019 IEEE/AIAA 38th Digital Avionics Systems Conf. (DASC). pp. 1–7 (2019)
3. Çetin, E., Barrado, C., Pastor, E.: Explainability of deep reinforcement learning method with drones. In: 2023 IEEE/AIAA 42nd Digital Avionics Systems Conf. (DASC). pp. 1–9 (2023)
4. Darwish, A.A., Nakhmani, A.: Drone navigation and target interception using deep reinforcement learning: A cascade reward approach. IEEE Journal of Indoor and Seamless Positioning and Navigation **1**, 130–140 (2023)
5. Deng, Z., Jiang, J., Long, G., Zhang, C.: Causal reinforcement learning: A survey (2023), https://arxiv.org/abs/2307.01452
6. Diehl, M., Ramirez-Amaro, K.: Why did i fail? a causal-based method to find explanations for robot failures. IEEE Robotics and Automation Letters **7**(4), 8925–8932 (2022)
7. Feliciano-Avelino, I., Méndez-Molina, A., Morales, E.F., Sucar, L.E.: Causal based action selection policy for reinforcement learning. In: Batyrshin, I., Gelbukh, A., Sidorov, G. (eds.) Advances in Computational Intelligence. pp. 213–227. Springer, Cham (2021)
8. García, M., Caballero, R., González, F., Viguria, A., Ollero, A.: Autonomous drone with ability to track and capture an aerial target. In: 2020 Int. Conf. on Unmanned Aircraft Systems (ICUAS). pp. 32–40 (2020)
9. Gatteschi, V., Lamberti, F., Paravati, G., Sanna, A., Demartini, C., Lisanti, A., Venezia, G.: New frontiers of delivery services using drones: A prototype system exploiting a quadcopter for autonomous drug shipments. In: 2015 IEEE 39th Annual Computer Software and Applications Conf. vol. 2, pp. 920–927 (2015)

10. Gonzalez-Soto, M., Sucar, L.E., Escalante, H.J.: Playing against nature: causal discovery for decision making under uncertainty (2018), https://arxiv.org/abs/1807.01268
11. Hanover, D., Loquercio, A., Bauersfeld, L., Romero, A., Penicka, R., Song, Y., Cioffi, G., Kaufmann, E., Scaramuzza, D.: Autonomous drone racing: A survey. IEEE Transactions on Robotics **40**, 3044–3067 (2024)
12. Hasan, K.M., Abdullah-Al-Nahid, Alim, M.A., Maniruzzaman, M., Atiqur Rahaman, G.M., Ahsan, M.S., Newaz, S.S.: Design and development of an aircraft type multi-functional autonomous drone. In: 2020 IEEE Region 10 Symposium (TENSYMP). pp. 734–737 (2020)
13. Heyn, H.M., Knauss, E.: Structural causal models as boundary objects in ai system development. In: 2022 IEEE/ACM 1st Int. Conf. on AI Engineering – Software Engineering for AI (CAIN). pp. 43–45 (2022)
14. Kersandt, K., Muñoz, G., Barrado, C.: Self-training by reinforcement learning for full-autonomous drones of the future. In: 2018 IEEE/AIAA 37th Digital Avionics Systems Conf. (DASC). pp. 1–10 (2018)
15. Klecka, W.: Discriminant Analysis. SAGE Publications, Inc. (1980). https://doi.org/10.4135/9781412983938
16. Koller, D., Friedman, N.: Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning. The MIT Press (2009)
17. Lewis-Beck, M.: Applied Regression. SAGE Publications, Inc. (1980)
18. Mario Silvagni, Andrea Tonoli, E.Z., Chiaberge, M.: Multipurpose uav for search and rescue operations in mountain avalanche events. Geomatics, Natural Hazards and Risk **8**(1), 18–33 (2017)
19. Méndez-Molina, A., F. Morales, E., Sucar, L.E.: Causal discovery and reinforcement learning: A synergistic integration. In: Salmerón, A., Rumí, R. (eds.) Proceedings of The 11th Int. Conf. on Probabilistic Graphical Models. Proceedings of Machine Learning Research, vol. 186, pp. 421–432. PMLR (05–07 Oct 2022)
20. Méndez-Molina, A., Morales, E.F., Sucar, L.E.: Carl: A synergistic framework for causal reinforcement learning. IEEE Access **11**, 126462–126481 (2023)
21. Pitis, S., Creager, E., Mandlekar, A., Garg, A.: Mocoda: Model-based counterfactual data augmentation (2022), https://arxiv.org/abs/2210.11287
22. Radoglou-Grammatikis, P., Sarigiannidis, P., Lagkas, T., Moscholios, I.: A compilation of uav applications for precision agriculture. Computer Networks **172**, 107148 (2020)
23. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: 2016 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). pp. 779–788 (2016)
24. Shi, W., Huang, G., Song, S., Wu, C.: Temporal-spatial causal interpretations for vision-based reinforcement learning. IEEE Transactions on Pattern Analysis and Machine Intelligence **44**(12), 10222–10235 (2022)
25. Shin, S.Y., Kang, Y.W., Kim, Y.G.: Automatic drone navigation in realistic 3d landscapes using deep reinforcement learning. In: 2019 6th Int. Conf. on Control, Decision and Information Technologies (CoDIT). pp. 1072–1077 (2019)
26. Sucar, L.E.: Probabilistic graphical models. Advances in computer vision and pattern recognition, Springer Nature, Cham, Switzerland, 2 edn. (Dec 2020)
27. Zeng, Y., Cai, R., Sun, F., Huang, L., Hao, Z.: A survey on causal reinforcement learning (2023), https://arxiv.org/abs/2302.05209
28. Zhu, W., Yu, C., Zhang, Q.: Causal deep reinforcement learning using observational data. In: Proceedings of the Thirty-Second Int. Joint Conf. on AI. p. 4711–4719. IJCAI-2023, Int. Joint Conf. on AI Organization (Aug 2023)