Contents lists available at ScienceDirect

Reliability Engineering and System Safety

journal homepage: www.elsevier.com/locate/ress

Hierarchical reinforcement learning for transportation infrastructure maintenance planning

Zachary Hamida [*], James-A. Goulet

Department of Civil, Geological and Mining Engineering, Polytechnique Montreal, 2500 Chem. de Polytechnique, Montreal, H3T 1J4, Quebec, Canada

ARTICLE INFO

Dataset link: https://github.com/CivML-PolyMtl/InfrastructuresPlanner

Keywords:
Maintenance planning
Reinforcement learning
RL environment
Deep Q-learning
Infrastructure deterioration
State-space models

ABSTRACT

Maintenance planning on bridges commonly faces multiple challenges, mainly related to complexity and scale. Those challenges stem from the large number of structural elements in each bridge in addition to the uncertainties surrounding their health condition, which is monitored using visual inspections at the element-level. Recent developments have relied on deep reinforcement learning (RL) for solving maintenance planning problems, with the aim to minimize the long-term costs. Nonetheless, existing RL based solutions have adopted approaches that often lacked the capacity to scale due to the inherently large state and action spaces. The aim of this paper is to introduce a hierarchical RL formulation for maintenance planning, which naturally adapts to the hierarchy of information and decisions in infrastructure. The hierarchical formulation enables decomposing large state and action spaces into smaller ones, by relying on state and temporal abstraction. An additional contribution from this paper is the development of an open-source RL environment that uses state-space models (SSM) to describe the propagation of the deterioration condition and speed over time. The functionality of this new environment is demonstrated by solving maintenance planning problems at the element-level, and the bridge-level.

1. Introduction

Transportation infrastructure such as roads, tunnels and bridges are continuously deteriorating due to aging, usage and other external factors [1]. Accordingly, maintenance planning for the aforementioned infrastructure aims at minimizing maintenance costs, while sustaining a safe and functional state for each structure [2,3]. Maintenance strategies for bridges can be either classified as time-based maintenance, such as recurring maintenance actions based on a fixed time interval, or condition-based maintenance (CBM) [4]. In the context of CBM, the main components involved in the development of any maintenance policy are quantitative measures for, (1) the structural health condition, (2) the effects of interventions, and (3) costs of maintenance actions. The structural health of bridges is commonly evaluated using visual inspections at the element-level [5–7]. An example of an element in this context is the *pavement* in a concrete bridge. The information at the element-level are thereafter aggregated to provide a representation for the overall deterioration state of a bridge [8]. Similarly, maintenance actions are performed at the element-level, and their corresponding effect is aggregated at the bridge-level [8,9]. The hierarchical nature of condition assessments, and maintenance actions presents challenges in formulating the bridge maintenance planning problem. First, the aggregation of the health states from the element-level to the bridge-level results in additional uncertainties, which render deterministic deterioration models insufficient [8]. Second, performing actions at the element-level implies that a decision-making framework is required to search for maintenance policies at the element-level in each bridge. Thus, the search-space for an optimal maintenance policy is typically large as it is common for a bridge to have hundreds of structural elements [10].

Existing approaches for solving the maintenance planning problem have adopted Markov decision process (MDP) formulations [2,3,11,12], relying on discrete states where transitioning from one state to another depends only on the current state [13]. The MDP approach is well-suited for small state-space problems, so that using MDP in the context of maintenance planning have incurred simplifications on the state and the action space [2,14]. An example of simplification is reducing the search space by merging the state representation of structural elements with similar deterioration states and maintenance actions [14].

The large state space has also motivated the application of reinforcement learning (RL) methods to search for optimal maintenance policies [2,15,16]. Conventional RL methods are well-suited for discrete action and state spaces, where the agent (or decision-maker) performs different actions and receives a feedback (rewards), which

* Corresponding author.
E-mail address: zachary.hamida@polymtl.ca (Z. Hamida).

https://doi.org/10.1016/j.ress.2023.109214
Received 8 October 2022; Received in revised form 26 February 2023; Accepted 1 March 2023
Available online 8 March 2023

can be used to update the value function corresponding to the visited states [13]. Existing work in the context of maintenance planning have focused mainly on multi-agent RL (MARL) methods, due to their compatibility with large action spaces [1,17–19]. Applications include coordinated reinforcement learning (CRL) for joint decision-making of multi-agents [18], and hierarchical RL where higher-level agents moderate the behavior of lower-level agents [19,20]. This latter application has been further improved by combining the CRL with the hierarchical reinforcement learning framework [17]. Despite the MARL extension, RL methods are inherently limited to low-dimensional problems and lack the capacity to scale for large state spaces without increasing the number of agents acting on the state space [2]. Accordingly, deep reinforcement learning (DRL) and multi-agent DRL have been proposed as an alternative due to their capacity of handling large and continuous state and action spaces. Specifically, frameworks with centralized training such as the branching dueling Q network (BDQN), deep centralized multi-agent actor–critic (DCMAC) and the advantage actor–critic (MAA2C) [2,21,22]. A common limitation associated with the aforementioned frameworks is the instability of the training and convergence, especially as the size of the action space increases [23]. In addition, the policy obtained is not interpretable, where it is not possible to plot the decision boundaries of the optimal policy. The interpretability is important due to the lack of a clear stopping criteria for training the agents in the context of planning problems, which makes it difficult to evaluate the validity of the reward function or the policy in practical applications. Another common limitation in the context of maintenance planning for transportation infrastructure is the use of discrete Markov models (DMM) for modeling the deterioration process over time. The use of the DMM framework in this context induces drawbacks related to overlooking the uncertainty associated with each inspector, and the incapacity to estimate the deterioration speed [5].

The aim of this paper is to introduce a hierarchical reinforcement learning formulation that adapts to the hierarchical nature of information in the maintenance planning problem. The hierarchical formulation enables decomposing large state and action spaces into smaller ones, by relying on state and temporal abstraction [24,25]. State abstraction enables representing the state-space of the planning problem by a hierarchy of states, such as, the element-level, the structural category level, and the bridge-level. Each of the aforementioned levels has an action-space, where interdependent policies can be learned and applied. Take for example, a bridge-level decision, where the action-space is defined as maintain or do nothing; if a policy suggests doing nothing, then no intervention is applied on all elements within the bridge, without assessing their health states.

The main contributions in this paper are: (1) formulating a hierarchical deep reinforcement learning approach that adapts to bridge maintenance planning, and provides advantages in scalability, and interpretability through visualizing the decision boundaries of the policies. (2) Incorporating the deterioration speed alongside the deterioration condition in the decision-making analyses [5]. (3) Developing a standard gym-based RL environment (link: https://github.com/CivML-PolyMtl/InfrastructuresPlanner) for emulating the deterioration process of bridges, based on state-space models (SSM) [8,26,27].

The performance of the proposed hierarchical approach is demonstrated using an example application for a bridge from the network of bridges in the province of Quebec, Canada. The analyses include a comparison with the BDQN and MAA2C approaches for planning maintenance on a multi-component system, in addition to learning a bridge-level maintenance policies.

The paper is organized as follows: Section 1.1, outlines the formulation of the maintenance planning problem in the context of transportation infrastructure. Section 2, provides a theoretical background for solving sequential decision-making problems, and Section 3, presents the proposed maintenance planning approach, in addition to providing details about the RL environment. The performance of the proposed maintenance planning approach is demonstrated in Section 4, which is followed by a discussion about its advantages and limitations. Finally, the conclusion of this work is provided in Section 5.

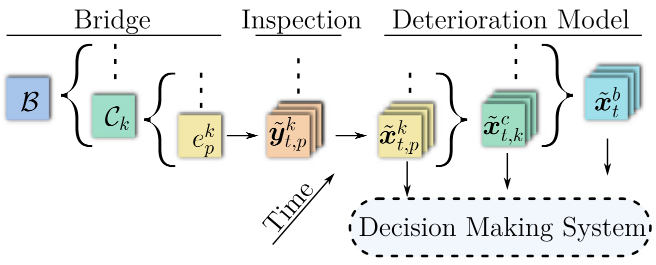

Bridge
Inspection
Deterioration Model
$\mathcal{B}$
$\mathcal{C}_k$
e_p^k
$\tilde{\boldsymbol{y}}_{t,p}^k$
$\tilde{\boldsymbol{x}}_{t,p}^k$
$\tilde{\boldsymbol{x}}_{t,k}^c$
$\tilde{\boldsymbol{x}}_t^b$
Time
Decision Making System

Fig. 1. Hierarchy of components and information in a bridge $\mathcal{B}$, where each structural category C_k is composed of a number of structural elements e_p^k. The element-level inspection data $\tilde{y}_{t,p}^k$ provides information about the health states at the element-level $\tilde{x}_{t,p}^k$, structural category level $\tilde{x}_{t,k}^c$, and bridge-level $\tilde{x}_t^b$.

1.1. Problem formulation

A bridge $\mathcal{B}$ from the network of bridges in the Quebec province is considered to demonstrate the decision-making analyses presented in this paper. Fig. 1 provides a summary for the hierarchy of components and information in bridge $\mathcal{B}$. The bridge is composed of $\mathtt{K}$ structural categories, each of which is composed of $\mathtt{P}$ structural elements. An example of a structural category is the *beams* category, which is the kth category in bridge $\mathcal{B}$ with a total of $\mathtt{P}$ beams as in, $C_k = \{e_1^k, \dots, e_p^k, \dots, e_{\mathtt{P}}^k\}$. Visual inspections, represented by $\tilde{y}_{t,p}^k$, are performed on the elements of bridge $\mathcal{B}$ every three years to monitor their health condition. The health states of the elements are denoted by $\tilde{x}_{t,p}^k$, and are inferred based on the inspection data $\tilde{y}_{t,p}^k$ which are defined in a continuous bounded space $[l,u] = [25,100]$; where $l = 25$ refers to a poor health condition and $u = 100$ refers to a perfect health condition. The element-level health states $\tilde{x}_{t,p}^k$ are thereafter aggregated for each structural category to provide the overall health states of the structural categories $\tilde{x}_{t,k}^c$. Similarly, the overall health states of the bridge $\tilde{x}_t^b$ are based on the aggregation of the health states from the structural categories [9]. It should be noted that for all the aforementioned levels, the health states are described by the same condition range defined by, $\tilde{x}_{t,p}^k, \tilde{x}_{t,k}^c, \tilde{x}_t^b \in [l,u]$, and the deterioration speed which is defined in $\mathbb{R}^-$. The $\tilde{}$ in $\tilde{x}_{p,t}^k$ refers to variables within the bounded space $[l,u] = [25,100]$, while the absence of $\tilde{}$ refers to the variables defined in the unbounded space $[-\infty,\infty]$ [9]. An example of a perfect health state is when the condition is, $\tilde{x}_t = 100$, and the deterioration speed is near-zero. Bridge $\mathcal{B}$ mainly undergo imperfect maintenance actions at the element-level, where the actions are represented by the set $\mathcal{A}^e = \{a_0, a_1, a_2, a_3, a_4\}$, with a_0: do nothing, a_1: routine maintenance, a_2: preventive maintenance, a_3: repair, and a_4: replace [10]. Each action is associated with a cost, in addition to other costs related to the service interruption and penalties for reaching a critical state.

2. Background

This section provides the theoretical background for the main concepts related to proposed decision making framework.

2.1. Markov decision processes (MDP)

A MDP is an approach to describe sequential decision-making problems using the tuple $\langle \mathcal{S}, \mathcal{A}, P, \mathcal{R} \rangle$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, P is the transition function, and $\mathcal{R}$ is the set of rewards [13]. Taking an action $a \in \mathcal{A}$ results in a transition from the state $s_t = s$ at time t, to the state $s_{t+1} = s'$ using a Markovian transition function $P(s'|s,a)$, which implies that the next state is only conditional on the pair of the current state s and action a. Each action $a \in \mathcal{A}$ taken in the MDP can affect the expected immediate reward r_t and the total rewards G_t [13]. In this context, the effect of an action can be either deterministic and accordingly the MDP is considered deterministic where, $\Pr(s'|s,a) = 1$, or otherwise, the MDP is considered stochastic when $\Pr(s'|s,a) \neq 1$ [28]. Similarly, states can be either represented by deterministic exact information (i.e., true state s) in the case of a MDP, or inferred information (i.e., belief about the true state s) in the case of a partially observed MDP (POMDP) [13].

A policy π in the context of MDP represents a mapping between states and actions, where a deterministic policy provides deterministic actions such that, $\pi(s) : s \to a$, while a stochastic policy provides probabilities for taking each action in each state $\pi(a|s) : s \to \Pr(a)$ [13]. Following a policy $\pi(\cdot)$ in a MDP would yield a total return at time t defined by,

$$G_t = \sum_{i=0}^{\infty} \gamma^i r(s_{t+i}, a_{t+i}), \tag{1}$$

where γ is the discount factor that enables considering an infinite planning horizon when $\gamma \in]0,1[$, and $r(s_t, a_t) = \mathbb{E}[R_t|S_t = s, A_t = a]$ denotes the expected reward given the state $S_t = s$ and the action $A_t = a$ [13]. In this context, the rewards represent a feedback associated with action a taken at each state s. Provided the notion of rewards driven by actions, evaluating a policy π is possible by using a value function $V_\pi(s)$ and an action-value function $Q_\pi(s,a)$. The value function represents the expected discounted return for being in a state s, under policy π, such that,

$$V_\pi(s) = \mathbb{E}_\pi\left[G_t|S_t = s\right] = r(s_t, a_t) + \mathbb{E}_\pi\left[\sum_{i=1}^{\infty} \gamma^i r(s_{t+i}, a_{t+i})|S_t = s\right], \tag{2}$$

where $\mathbb{E}_\pi$ is the expected value while following the policy π. On the other hand, the action-value function $Q_\pi(s,a)$ refers to the expected discounted return for taking an action a, in a state s, based on policy π, which is described by,

$$Q_\pi(s,a) = r(s_t, a_t) + \mathbb{E}_\pi\left[\sum_{i=1}^{\infty} \gamma^i r(s_{t+i}, a_{t+i})|S_t = s, A_t = a\right]. \tag{3}$$

Accordingly, a policy π_* is considered optimal when the state-value function and action-value function are,

$$\begin{aligned} V_*(s_t) &= \max_\pi V_\pi(s_t),\ \forall s_t \in \mathcal{S}, \\ Q_*(s_t, a_t) &= \max_\pi Q_\pi(s_t, a_t),\ \forall s_t \in \mathcal{S}, a_t \in \mathcal{A}. \end{aligned} \tag{4}$$

2.2. Semi-Markov decision process

A semi-Markov decision process (SMDP) formulation is similar to a MDP, with the exception that a SMDP considers actions to have a duration $\bar{\mathrm{T}}$ to be performed [28]. An example for a SMDP action is the task of maintaining a bridge, which requires a duration $\bar{\mathrm{T}}$, to perform the maintenance actions for each element within the bridge. From this example it can be inferred that actions (or tasks) in the SMDP are performed at different levels (i.e., element-level and bridge-level). The expected rewards $\bar{r}(s_t, a_t^\ell)$ associated with the task a_t^ℓ at level ℓ are estimated using,

$$\bar{r}(s_t, a_t^\ell) = \mathbb{E}_{\pi^{\ell-1}}\left[\sum_{i=0}^{\bar{\mathrm{T}}} \gamma^i r(s_{t+i+1}, a_{t+i+1}^{\ell-1})|S_t = s, a_t^{\ell-1} = \pi^{\ell-1}(s_t)\right], \tag{5}$$

where $\bar{r}(s_t, a_t^\ell)$ is the expected cumulative discounted reward while following the policy $\pi^{\ell-1}$ from time t until the termination of the task a_t^ℓ after $\bar{\mathrm{T}}$ time-steps. Based on Eq. (11), the application of the SMDP formulation generally relies on state and temporal abstractions [24,25, 28]. The aim of state abstraction is to reduce the state space by aggregating states having similar properties without changing the essence of the problem [24,25]. This implies the feasibility of mapping a state $s \in \mathcal{S}$ to an abstract state $s_\phi \in \mathcal{S}_\phi$ while maintaining a near-optimal policy search, where the space $\mathcal{S}_\phi$ has a fewer states (i.e., $|\mathcal{S}_\phi| \ll |\mathcal{S}|$) [24]. Fig. 2 shows an illustrative example, where the real state is represented by different levels of abstraction. On the other hand, a temporal abstraction is applied when actions are taking place at

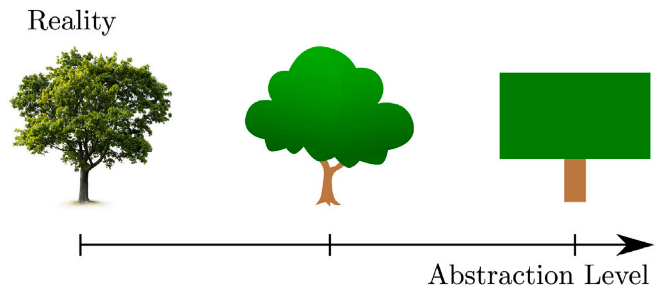


Fig. 2. Illustrative example showing two levels of abstraction starting from the state of reality on the left.

different time scales [13]. For example, applying an intervention on a bridge b_j from time t to time $t+1$, involves many actions performed at the element-level over a sub-timestamps τ, such that, $t < (t+\tau) < t+1$.

2.3. Deep reinforcement learning

Typical RL approaches rely on interactions between a decision maker (the agent) and an environment in order to learn a policy that maximizes the total cumulative rewards. A common technique for learning from interactions is the use of the temporal difference (TD) [29], to perform recursive updates on the action-value function $Q(s_t, a_t)$ such that,

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \eta\left[r(s_t, a_t) + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\right], \tag{6}$$

where η denotes the learning rate. Updating the $Q(s_t, a_t)$ function using Eq. (6) requires a table for all pairs of states and actions, which can be challenging for large and continuous state and action spaces [13]. Therefore, Deep RL methods have provided a scalable alternative to the tabular Q-learning, which enables approximating the optimal action-value function such that, $Q(s,a;\boldsymbol{\theta}) \approx Q^*(s,a)$. Similar to Eq. (6), deep RL relies on temporal difference (TD) to estimate the set of parameters $\boldsymbol{\theta}$ using,

$$\mathcal{L}_i(\boldsymbol{\theta}_i) = \mathbb{E}\left[r + \gamma \max_{a'} Q(s', a'; \boldsymbol{\theta}^-) - Q(s, a; \boldsymbol{\theta}_i)\right]^2, \tag{7}$$

where $\mathcal{L}_i(\boldsymbol{\theta}_i)$ is the loss function associated with the parameters $\boldsymbol{\theta}_i$, and $\boldsymbol{\theta}^-$ represents the parameters of the target model, which is a delayed replica of the $Q(\cdot)$ function. The role of a target model is to stabilize the learning process where the parameters of the target model $\boldsymbol{\theta}^-$ are updated based on $\boldsymbol{\theta}_i$ by using a soft update approach [30]. Eq. (7) is the foundation for many different DRL methods for identifying an optimal policy, nonetheless, the choice of an approach mainly depends on the design and properties of the MDP [13].

2.4. Hierarchical reinforcement learning

Hierarchical RL enables applying the principles of RL on SMDP environments, where there are multiple tasks occurring simultaneously at different time scales [28]. The hierarchy here refers to multiple layers of policies, where the higher level policy dictates the behavior of the lower level policies [31]. For example, the higher level policy observes the overall deterioration state of a bridge $\tilde{x}_t^b$ and provides a target state $\tilde{x}_t^b + \delta^b$, which the lower level policies will try to match by observing and acting on the deterioration states of the structural elements. Based on the definition above and Eq. (5), the action-value function $Q(s_t, a_t^\ell)$ for an optimal policy can be defined as,

$$Q(s_t, a_t^\ell) = \bar{r}(s_t, a_t^\ell) + \sum_{s_{t+\bar{\mathrm{T}}}} \sum_{\bar{\mathrm{T}}} \gamma^{\bar{\mathrm{T}}} P(s_{t+\bar{\mathrm{T}}}, \bar{\mathrm{T}}|s_t, a_t^\ell) \max_{a_{t+\bar{\mathrm{T}}}^\ell} Q(s_{t+\bar{\mathrm{T}}}, a_{t+\bar{\mathrm{T}}}^\ell). \tag{8}$$

From Eq. (8), the transition model $P(s_{t+\bar{\mathrm{T}}}, \bar{\mathrm{T}}|s_t, a_t^\ell)$ and the reward $\bar{r}(s_t, a_t^\ell)$ depend directly on the subsequent policy $\pi^{\ell-1}$ [28].

Learning the hierarchical policies can be done by using either an end-to-end approach where all policies are trained simultaneously, or a bottom-to-top approach starting from the lower level policies [28,32]. The latter approach is favored for large-scale problems provided the instability issues for centralized joint training of multiple policies [32].

3. Hierarchical deep RL for bridge maintenance planning

Fig. 3 shows an illustration for the hierarchical maintenance planning architecture, where the state of the environment at time t is represented using different levels: a bridge level with state s^b_t, a structural-category level with $s^c_{t,k}$, and an element-level with $s^e_{t,p}$. Each of the aforementioned states provide information about the health of the bridge at its corresponding level. For example, the state of each element $s^e_{t,p}$ contains information about the deterioration condition $\tilde{x}^k_{t,p}$ and speed $\dot{\tilde{x}}^k_{t,p}$ of the pth structural element e^k_p.

The hierarchical framework is composed of a centralized agent for the bridge level with policy π^b, and decentralized agents for each structural category represented by the policy π_k. The centralized agent proposes a target improvement $\delta^b \leftarrow \pi^b(s^b_t)$ for the health condition of the bridge x^b_t, such that the health condition of the bridge at time $t+1$ is $x^b_{t+1} + \delta^b$. If the improvement value $\delta^b = 0$, then no maintenance is applied on the bridge; otherwise, maintenance actions are performed according to the improvement value δ^b, defined within, $\delta^b \in [0, (u-l)]$, where l is the lower bound and u is the upper bound for the condition.

As shown in Fig. 3, the hierarchical framework aims to decode the bridge-level target improvement δ^b to a vector of actions for all structural elements in $\mathcal{B}$. This can be achieved sequentially by distributing δ^b on the structural categories according to their current deterioration condition $\tilde{x}^c_{t,k}$ using,

$$\delta^c_k(\delta^b) = \frac{u - \tilde{x}^c_{t,k}}{u \cdot \mathtt{K} - \sum_{k=1}^{\mathtt{K}} \tilde{x}^c_{t,k}} \cdot \mathtt{K} \cdot \delta^b, \tag{9}$$

where δ^c_k is the target improvement for the kth structural category C_k, $\mathtt{K}$ is the total number of structural categories within the bridge, and u is the perfect condition. From Eq. (9), if $\delta^c_k > 0$, then the structural element $e^k_p \in C_k$ is maintained according to the policy π_k. Thereafter, the states of the structural category $\tilde{s}^c_{t,k}$, and the bridge $\tilde{s}^b_t$ are updated with the state after taking the maintenance action $a^k_{t,p} \leftarrow \pi_k(s^e_{t,p})$ on the structural element e^k_p. In order to determine if the next structural element $p+1$ requires maintenance, the target improvement δ^c_k and δ^b are updated using,

$$\begin{aligned}\delta^c_k &= \max\left(\tilde{x}^c_{t,k\,(\text{before maintenance})} + \delta^c_k - \tilde{x}^c_{t,k\,(\text{updated})}, 0\right),\\ \delta^b &= \max\left(\tilde{x}^b_{t\,(\text{before maintenance})} + \delta^b - \tilde{x}^b_{t\,(\text{updated})}, 0\right).\end{aligned} \tag{10}$$

Once the updated target improvement δ^c_k reaches $\delta^c_k = 0$, the remaining structural elements within C_k are assigned the action (a_0:*do nothing*). The aforementioned steps are repeated for each structural category C_k in bridge $\mathcal{B}$ until all elements e^k_p are assigned a maintenance action $a^k_p \in \mathcal{A}^e$.

The element-level actions are defined by the set $\mathcal{A}^e = \{a_0, a_1, a_2, a_3, a_4\}$, where a_0: do nothing, a_1: routine maintenance, a_2: preventive maintenance, a_3: repair, and a_4: replace [10]. The corresponding effect associated with each of the aforementioned actions is estimated using a data-driven approach [9]. Moreover, the cost associated with each element-level maintenance action is defined as a function of the deterioration state of the structural element, and for each structural category. Further details about the effect of interventions and maintenance costs are provided in Appendix B.3.

In addition to the maintenance action costs, there are costs related to the bridge service-stoppage and penalties for reaching a critical state. The service-stoppage costs are defined to prevent frequent interruptions for the bridge service, as well as to encourage performing all of the required maintenance actions at the same time. On the other hand, the penalties are applied when a predefined critical state is reached and no maintenance action is taken. The critical state in this work is defined in accordance with the definition provided by the *Manual of Inspections* [10], for a deterioration state that requires maintenance.

3.1. Learning the policies in the hierarchical DRL

Learning the policies in the hierarchical DRL framework is done using a bottom-to-top approach starting from the element-level policies, and by relying on decentralized element-level agents with a centralized bridge-level agent [28,32]. Such an approach offers flexibility in using transfer learning for structural elements that share similar properties. In this context, structural elements from a same structural category (e.g., all the beams) are assumed to have a similar deterioration dynamics and similar cost function for maintenance actions. Therefore, the number of element-level agents that require training is equivalent to the number of structural categories in bridge $\mathcal{B}$.

Training the element-level agents is done based on a MDP environment that mimics the deterioration process and provides information about the deterioration condition $\tilde{x}^k_{t,p}$ and speed $\dot{\tilde{x}}^k_{t,p}$ of the structural elements (see Section 3.2). Accordingly, the state space for the element level is, $s^e_t = [\tilde{x}^k_{t,p}, \dot{\tilde{x}}^k_{t,p}]$ and the action space is defined by the set $\mathcal{A}^e$. Training the element-level agents can be done using off-policy methods, such as deep Q-learning with experience replay [13].

After learning the policies $\pi_{1:\mathtt{K}}$, it becomes possible to learn the centralized bridge-level policy, which observes the state $s^b_t = [\tilde{x}^b_t, \dot{\tilde{x}}^b_t, \sigma^b_t]$, where $\tilde{x}^b_t$ is the overall health condition of the bridge, $\dot{\tilde{x}}^b_t$ is the overall deterioration speed of the bridge, and σ^b_t is the standard deviation for the condition of the structural categories in the bridge $\sigma^b_t = \text{std.}(\tilde{\boldsymbol{x}}^c_{t,1:\mathtt{K}})$. The environment at the bridge level is a SMDP due to the assumption that all element level maintenance actions are occurring between the time steps t and $t+1$. Training the centralized agent is done using an off-policy deep Q-learning approach with experience replay. The bridge-level agent experience transition is composed of, $(s^b_t, \delta^b_t, r^b_t, s^b_{t+1})$, where r^b_t is the total costs from all actions performed on the bridge and is defined by,

$$r^b(s^b_t, \delta^b_t) = r^s + \sum_{k=1}^{\mathtt{K}} \sum_{p=1}^{\mathtt{P}} r(s^e_{t,p}, a^k_{t,p}). \tag{11}$$

From Eq. (11), r^s is the service-stoppage cost for performing the maintenance actions. The next section describes the environment utilized for emulating the deterioration of bridges over time.

3.2. Deterioration state transition

The RL environment is built based on the deterioration and intervention framework developed by Hamida and Goulet [8,9], and is calibrated using the inspections and intervention database for the network of bridges in the Quebec province, Canada. The environment emulates the deterioration process by generating true states for all the elements e^k_p, using the transition model,

$$\overbrace{\boldsymbol{x}^k_{t,p} = \boldsymbol{A}_t \boldsymbol{x}^k_{t-1,p} + \boldsymbol{w}_t}^{\text{transition model}}, \underbrace{\boldsymbol{w}_t : \boldsymbol{W} \sim \mathcal{N}(\boldsymbol{w}; \boldsymbol{0}, \boldsymbol{Q}_t)}_{\text{process errors}}, \tag{12}$$

where $\boldsymbol{x}^k_{t,p} : \boldsymbol{X} \sim \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$ is a hidden state vector at time t associated with the element e^k_p. The hidden state vector $\boldsymbol{x}^k_{t,p}$ is a concatenation of the states that represent, the deterioration condition $x^k_{t,p}$, speed $\dot{x}^k_{t,p}$, and acceleration $\ddot{x}^k_{t,p}$, as well as the improvement due to interventions represented by, the change in the condition $\delta^e_{t,p}$, the speed $\dot{\delta}^e_{t,p}$, and the acceleration $\ddot{\delta}^e_{t,p}$. $\boldsymbol{A}_t$ is the state transition matrix, and $\boldsymbol{w}_t$ is the process error with covariance matrix $\boldsymbol{Q}_t$. Eq. (12) represents the dynamics of a transition between the states in the context of a MDP. In order to emulate the uncertainties about the deterioration state,

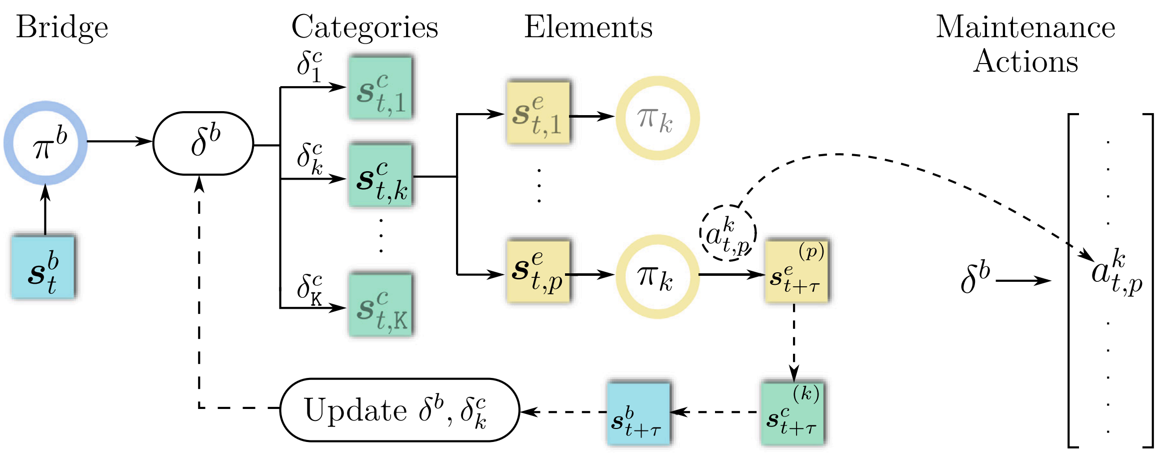
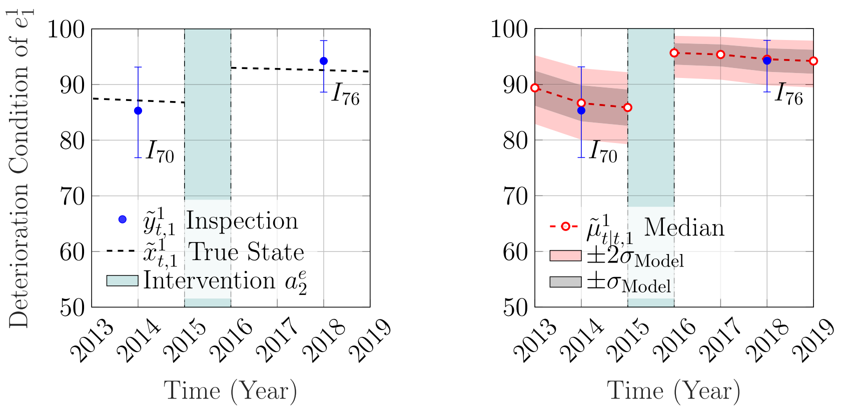


Fig. 3. Hierarchical deep RL for performing maintenance using a hierarchy of policies composed of, a centralized policy π^b for the bridge level, and decentralized element-level policies π_k. The centralized policy π^b produces a target improvement δ^b based on the bridge state s_t^b. The improvement δ^b is distributed on the structural categories to provide the category-wise improvements δ_k^c, which are sequentially translated to a vector of maintenance actions at the element-level using the policies π_k.



(a) Example for the true state of deterioration with synthetic inspections generated using the observation model, and an intervention at year 2016.

(b) Example for the KF forward inference based on the synthetic inspection data, with an intervention at year 2016.

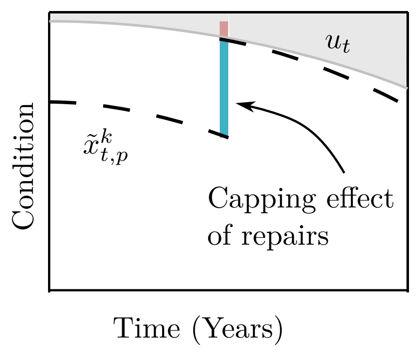
Fig. 4. Illustrative example for a deterministic deterioration curve (MDP environment) in Fig. 4(a), and an uncertain deterioration curve (POMDP environment) in Fig. 4(b).

synthetic inspection data $y_{p,t}^k$ are sampled at a predefined inspection interval using,

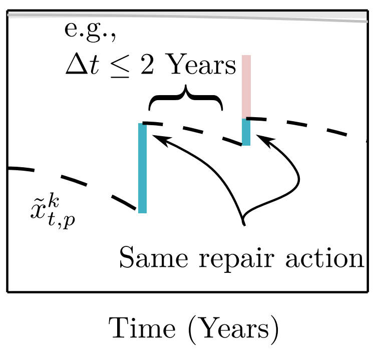
$$\overbrace{y_{t,p}^k = \boldsymbol{C}\boldsymbol{x}_{t,p}^k + v_t}^{\text{observation model}},\ \underbrace{v_t : V \sim \mathcal{N}(v;\mu_V(I_i),\sigma_V^2(I_i))}_{\text{observation errors}}, \tag{13}$$

where $\boldsymbol{C}$ is the observation matrix, and $v_t : V \sim \mathcal{N}(v;\mu_V(I_i),\sigma_V^2(I_i))$, is the observation error associated with each synthetic inspector $I_i \in \mathcal{I}$. The role of the synthetic inspection data is to provide imperfect measurements similar to the real world context. The information from this measurement at time t can be extracted using the Kalman Filter (KF), where the state estimates from the KF at each time t represent a belief about the deterioration state [33]. The state estimates from the KF provide a POMDP representation of the deterioration process. Fig. 4 shows an illustrative example for the deterioration and effect of interventions on a structural element. The true state $\tilde{x}_{t,1}^1$ in Fig. 4(a) is generated using the transition model in Eq. (12), while the synthetic inspections are generated using the observation model In Eq. (13). The KF inference is performed based on the synthetic inspections, and is represented by the expected value $\tilde{\mu}_{t|t,1}^1$ and the confidence region $\pm\sigma_{t|t}$ and $\pm 2\sigma_{t|t}$ shown in Fig. 4(b). The deterioration states $\tilde{\boldsymbol{x}}_{t,p}^k$ at the element-level are aggregated to obtain the overall deterioration state of the structural category $\boldsymbol{x}_{t,k}^c$, which are similarly aggregated to obtain the deterioration states estimates of the bridge $\boldsymbol{x}_t^b$. Further details about the aggregation procedure as well as the deterioration and interventions framework are provided in Appendix B.

Throughout the deterioration process, the effectiveness of repair actions is distinguished from the replacement action by introducing a decaying factor on the perfect state u_t, such that, $u_{p,t+1} = \rho_0 \times u_{p,t}$, where $0 < \rho_0 < 1$. This implies that repair actions are unable to restore a structural element to the original perfect health condition (i.e., $u_t = 100$) as it advances in age. Fig. 5(a) shows an illustration for the decay in the perfect condition u_t that can be reached by repair actions. Other practical considerations in this environment are related to capping the effect of maintenance after applying the same repair action repeatedly within a short period of time (e.g., $\Delta t \leq 2$). For example, if an action a_3 has an effect of $\delta_t^e = +20$, applying a_3 action two times in two consecutive years should not improve the structural element condition by, $\delta_t^e + \delta_{t+1}^e = 20 + 20$, but rather should improve the condition by, $\delta_t^e + \rho_1\delta_{t+1}^e$, where $0 < \rho_1 < 1$. Fig. 5(b) illustrates the capped effect of intervention caused by applying the same action twice within a short time interval. Further details about the choice of decaying factors are provided in Appendix B.2.



(a) Example for the capped effect of intervention due to the perfect state u_t (gray line) being decayed with the age of the structural element.



(b) Example for the capped effect of intervention due to applying the same repair action twice within a short period of time.

Fig. 5. Examples for scenarios where the effect of interventions is capped due to aging in Fig. 5(a), or repeatedly applying the same maintenance action as shown in Fig. 5(b).

4. Example of application

The performance of the proposed HRL framework is demonstrated on a case study for a bridge within the province of Quebec. Note that both the deterioration and interventions models are calibrated on data from the bridge network in the Quebec province, Canada [8].

4.1. Maintenance policy for a bridge with one structural category

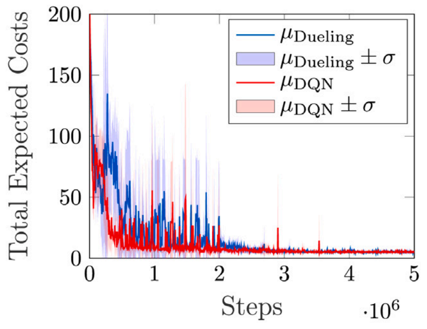
The goal in this example is to demonstrate the capacity of the HRL to achieve a near-optimal solution in a toy-problem, with a simple hierarchy of actions. In this context, the planning scope on bridge $\mathcal{B}$ considers only one structural category $\mathtt{K} = 1$, which corresponds to the *beams* structural category C_1. The beam elements in C_1 have a common critical deterioration condition $\tilde{x}_t$ and deterioration speed $\tilde{\dot{x}}_t$ defined as, $\tilde{x}_t = 55, \dot{x}_t = -1.5$. The aforementioned values are derived from the *manual of inspections* [10], and imply that the structural element requires a maintenance action when the critical state is reached; accordingly, taking no-action after reaching the critical state will incur a cost penalty on the decision-maker.

As described in Section 3.1, the first step to train the proposed hierarchical framework is to learn the task of maintaining beam structural elements at the element level. The policy π_k decides the type of maintenance actions based on the information about a deterministic deterioration condition $\tilde{x}^1_{t,p}$ and a deterministic deterioration speed $\tilde{\dot{x}}^1_{t,p}$ of the structural element such that, $s^e_{t,p} = [\tilde{x}^k_{t,p}, \tilde{\dot{x}}^k_{t,p}]$. The action set in this MDP is defined as, $\mathcal{A}^e = \{a_0, a_1, a_2, a_3, a_4\}$, which corresponds to a_0: do nothing, a_1: routine maintenance, a_2: preventive maintenance, a_3: repair, and a_4: replace [10]. The costs and effects associated with each of the aforementioned actions are described in Appendix B.3. Learning the maintenance policy π_k is done using a vectorized version of the RL environment which is detailed in Section 3.2 and Appendix A. The experimental setup for the training include a total of 5×10^4 episodes with the episode length defined as, $\mathtt{T} = 100$ years. The episode length is determined such that it is long enough to necessitate a replacement action, provided that the average life-span for a structural element is about 60 years [9]. Despite the fixed episode length, there is no terminal state as the planning horizon is considered infinite with a discount factor $\gamma = 0.99$. Moreover, the initial state in the RL environment is randomized where it is possible for a structural element to start the episode in a poor health state or a perfect health state. Fig. 6 shows the training and average performance for DQN and Dueling agents, along with two realizations for the optimal policy map obtained at the end of the training for each agent. The configuration for the DRL agents are provided in Appendix A. From Fig. 6(a), it is noticeable that the DRL agents reach a stable policy after 3×10^6 steps. Moreover, Fig. 6(b) shows optimal policy maps obtained by the DQN agent (left) and the Dueling agent (right), for the action space $\mathcal{A}^e$, with the critical state region highlighted by the area within the red boundary. From the policy maps, it can be noticed that the element's critical state region is dominated by major repairs, which is expected due to the penalties applied on the DRL agent if the structural element reaches that state. Despite the slight differences between the two optimal policy maps in Fig. 6(b), the policy map by the DQN agent is favorable because the action a_0 : *do nothing* did not leak into the predefined critical state region for the condition $\tilde{x}_t$ and speed $\tilde{\dot{x}}_t$. The leakage of the action a_0 : *do nothing* in the critical state region can occur due to interpolating the Q function values for states that are rarely visited by the agent, such as structural elements with a perfect condition $\tilde{x}_t = 100$, and high deterioration speed $\tilde{\dot{x}}_t = -1.6$.

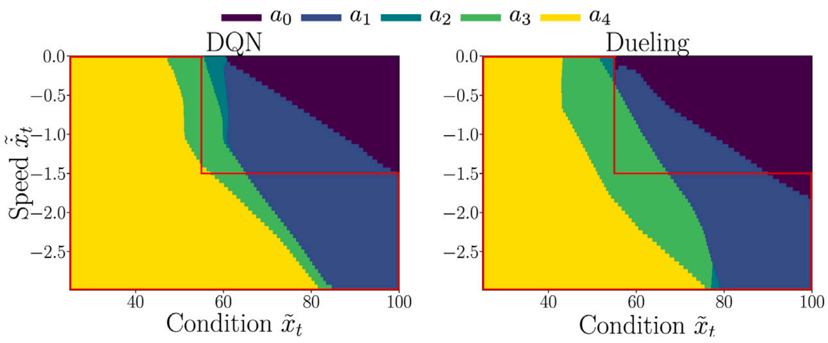
The optimal policy π^*_k provides the basis for decision-making at the bridge level, which corresponds to learning the bridge level policy π^b. The state-space is defined as, $s^b_t = [\tilde{x}^b_{t,1}, \tilde{\dot{x}}^b_{t,1}, \sigma^e_t]$, where $\tilde{x}^c_{t,1}, \tilde{\dot{x}}^1_{t,1}$ represent the overall deterioration condition and speed for the bridge, and σ^e_t is the standard deviation for the condition of the elements at each time step t. The action-space has one action δ^b, which corresponds to the target improvement, with $\delta^b = 0$ being equivalent to *do nothing*, and $0 > \delta^b \geq (u - l)$ is *maintain* the beam structural elements using π^*_k. Learning the policy π^b can be done using the vectorized RL environment at the bridge level, and the same DQN agent described in Appendix A. In this study, the continuous action space is discretized with $\delta^b = \{\delta^b_1, \ldots, \delta^b_{\mathtt{A}}\}$ to make it compatible with discrete action algorithms [34]. Accordingly, δ^b is represented by $\mathtt{A} = 10$ discrete actions equally spaced over its continuous domain.

In order to assess the scalability and performance of the proposed HRL framework, the total number of the structural elements in $C_1 \in \mathcal{B}$ is varied with $\mathtt{P} = \{5, 10, 15\}$ beam elements. The performance of the HRL framework is evaluated using 5 different environment seeds, and is compared against (1) the branching dueling DQN (BDQN) and (2) multi agent advantage actor–critic (MAA2C) [22]. The BDQN framework architecture, hyperparameters and configurations are adapted from Tavakoli et al. [21], while the configuration of MAA2C are derived from Papoudakis et al. [22], and are further described in Appendix A. Fig. 7 shows the results of the comparison based a structural category C_1 with $\mathtt{P} = 5$ beam elements in Fig. 7(a), a C_1 with $\mathtt{P} = 10$ beam elements in Fig. 7(b), and a C_1 with $\mathtt{P} = 15$ beam elements in Fig. 7(c).

From Fig. 7, the performance of the proposed HRL framework is reported while considering the pre-training phase required for learning the element level policy $\pi_{k=1}$, which extends over 3×10^6 steps. Based on the results shown in Fig. 7(a) for the case with $\mathtt{P} = 5$, the HRL and BDQN frameworks achieve a similar total expected rewards, however, the BDQN approach shows a faster convergence due to the end-to-end

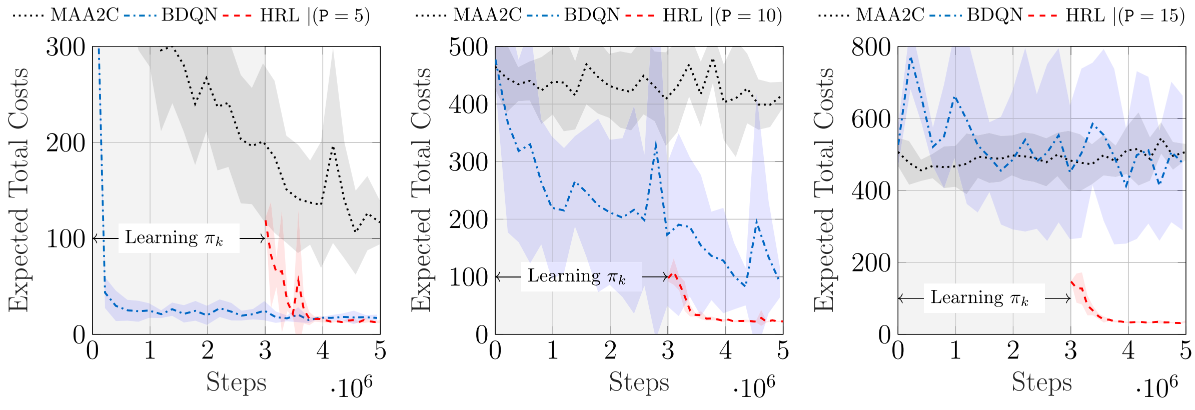


(a) Average performance based on 5 seeds for DQN and Dueling agents in learning the maintenance policy $\pi_{k=1}$ for a beam structural element.



(b) Two realizations for the optimal policy maps $\pi^*_{k=1}$ based on the DQN agent (left) and the Dueling agent (right), and according to the action space $\mathcal{A}^e$. The area within the red frame represents the predefined critical state region for the condition $\tilde{x}_t$ and speed $\dot{\tilde{x}}_t$.

Fig. 6. The training process of deep RL agents along with two realizations for the optimal policy $\pi^*_{k=1}$ of a beam structural element.



(a) (b) (c)

Fig. 7. Comparison between the proposed HRL, MAA2C and BDQN for learning the maintenance policy of a structural category C_1 with $\mathtt{P} = 5$ elements in Fig. 7(a), a C_1 with $\mathtt{P} = 10$ elements in Fig. 7(b), and a C_1 with $\mathtt{P} = 15$ elements in Fig. 7(c). The training results are reported based on the average performance on 5 seeds, with the confidence interval represented by $\pm\sigma$.

training. Nonetheless, when the number of beam elements increases in the case of $\mathtt{P} = 10$ and $\mathtt{P} = 15$, the HRL framework outperforms the BDQN and MAA2C approaches in terms of convergence speed and in the total expected return achieved in this experiment setup. This can be attributed to the BDQN considering each additional beam element as a distinct branch which leads to a significant increase in the size of the neural network model, and thus requiring a higher number of samples for the training. As for MAA2C, it suffers from an increase in the variance of the gradient estimates as the number of agents increases; which can affect the performance and stability of the method [23].

4.2. Maintenance policy for a bridge with multiple structural categories

This example extends the application of the proposed formulation to a planning problem involving $\mathtt{K} = 6$ structural categories within a bridge $\mathcal{B}$. Each structural category consists of a different number of structural elements which are summarized in Fig. 8. In this example, the bridge's health state is represented by, $\boldsymbol{s}^b = [\tilde{x}^b_t, \dot{\tilde{x}}^b_t, \sigma^c]$, where σ^c is the standard-deviation for the deterioration condition of the structural categories within $\mathcal{B}$. Similarly to the previous example, the bridge-level action is the target improvement δ^b which is represented by 10 discrete action bins uniformly covering the continuous domain. Solving this maintenance planning problem is done by first identifying the optimal policy for the structural elements e^k_p within each structural category C_k, where the element-level actions in each structural category have

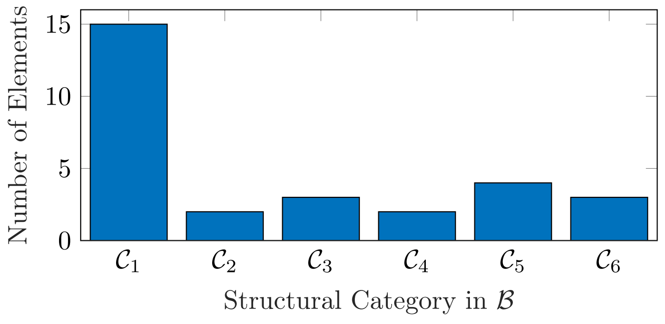


Fig. 8. Number of structural elements within each structural category in bridge $\mathcal{B}$, where C_1 represents the beams, C_2 is the front-walls, C_3 is the slabs, C_4 is the guardrail, C_5 is the wing-wall, and C_6 represents the pavement.

different costs and effects on the element's state (see Appendix B). Fig. 9 shows the optimal policy maps as learned by a DQN RL agent for the elements within each type of structural category. It should be noted that the characteristics of structural elements (e.g., critical thresholds) within each structural category are assumed to be identical, which allows learning a single policy per structural category. After obtaining the policies $\pi^*_{1:\mathtt{K}}$, the hierarchical RL framework is trained using the environment at the bridge level. The training process for the DQN agent

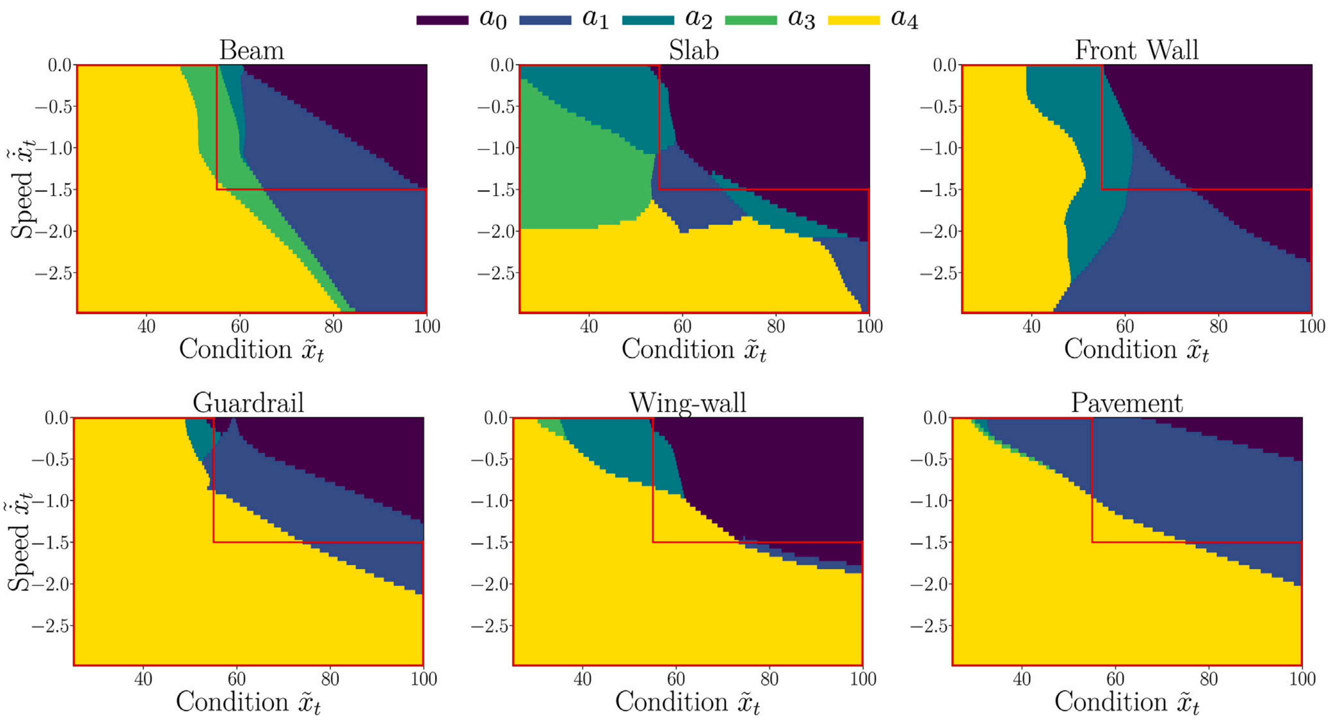


Fig. 9. The element-level policy maps for the action space $\mathcal{A}^e$, where each policy map is learned independently by a DQN agent.

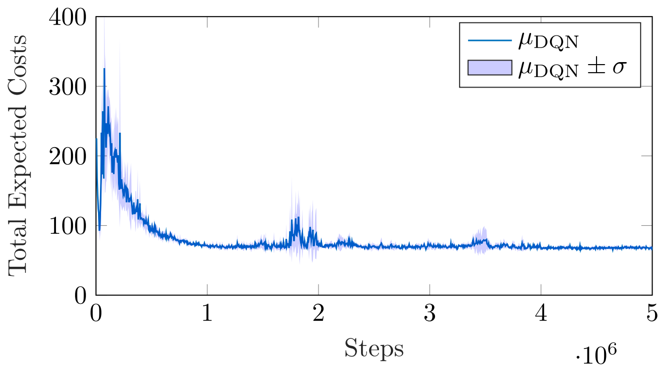


Fig. 10. Average performance based on 5 seeds for DQN agents in learning the maintenance policy π^b for a bridge composed of multiple structural categories.

at the bridge-level is reported using the average performance on 5 different seeds for the environment, as shown in Fig. 10, where it can be noticed that the policy's training became stable after 2×10^6 steps.

In order to use the hierarchical DRL agent for decision making on bridge $\mathcal{B}$, it is required to obtain the deterioration state estimates for each structural element, as well as the overall deterioration state of the bridge $\mathcal{B}$. This step can be achieved by relying on the element-level inspection data and using the SSM-based deterioration model for estimating and aggregating the deterioration states [8]. Accordingly, the policy π^b in the HRL framework relies on $\boldsymbol{s}^b = [\tilde{\mu}^b_t, \tilde{\dot{\mu}}^b_t, \sigma^c]$, where $\tilde{\mu}^b_t$ and $\tilde{\dot{\mu}}^b_t$ are the expected values for the deterioration condition and speed, respectively. On the other hand, each policy π_k depends on the expected values of the deterioration condition $\tilde{\mu}^k_{t,p}$ and speed $\tilde{\dot{\mu}}^k_{t,p}$ at the element level, as in $s^b_t = [\tilde{\mu}^k_{t,p}, \tilde{\dot{\mu}}^k_{t,p}]$. Fig. 11 illustrates the deterioration state estimates and the effect of maintenance on the deterioration condition and speed of the bridge $\mathcal{B}$. In this context, the decision making analyses are performed using a window of 10 years, starting from the year 2021.

The aggregated synthetic inspections illustrated in the magenta squares on Fig. 11 are generated using the mechanism described in Section 3.2 for generating synthetic data. From Fig. 11, and based on the bridge state $\boldsymbol{s}^b$, the HRL agent suggests to perform maintenance actions at years 2022 and year 2029. The breakdown for the maintenance actions at the element level is shown in Fig. 12, where the majority of the proposed maintenance actions are a_1 : routine maintenance with the exception to a wing-wall element that is suggested to undergo a replacement in the year 2022. The replacement action is suggested by the policy π_k mainly due to a high deterioration speed as $\tilde{\dot{\mu}}^k_{t,p} < -1.5$, which bypasses the critical state's speed threshold.

4.3. Discussion

Typical bridges have hundreds of structural elements, many of which exhibit a similar structural deterioration behavior. Capitalizing on the structural similarities can enable tackling maintenance planning for bridges at scale, in a sense that learning the task of maintaining a beam structural element, can be either generalized for other similar beam structural elements, or can provide a source policy that would accelerate the learning of maintenance tasks (e.g., maintaining slabs). The latter falls under transfer learning in RL [35], and is not covered in the scope of this work, yet it shows potential for future work. The proposed HRL approach offers the capacity to take advantage of the aforementioned aspects, in addition to providing an interpretable decision maps that enable verifying the coherence of the optimal maintenance policy. This is important because in the context of maintenance planning there is no clear definition for the stopping criteria during the training, unlike some classic RL benchmarks, where the stopping criteria can be defined based on the success rate of the agent in accomplishing the task.

The advantages in the proposed HRL coincide with limitations that are mainly related to state abstraction and learning the policies. The level of abstraction in representing the state is dependent on the number of structural elements in the bridge, such that, for a bridge with many structural elements $\mathtt{E} > 100$, the overall bridge condition and speed may not be sufficient to fully describe the health state, and accordingly the abstract state space should be augmented with additional information such as, the overall health condition and speed

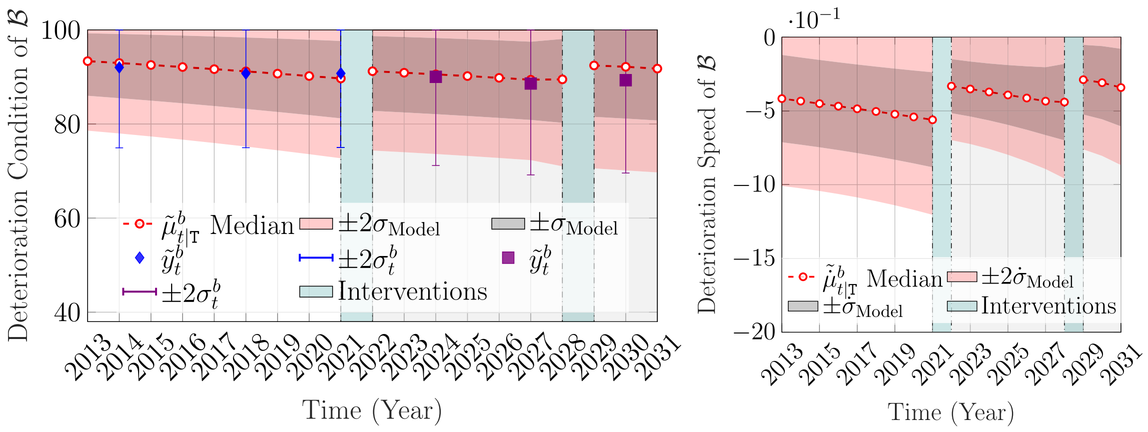


Fig. 11. Deterioration state estimates for the condition and the speed of bridge $\mathcal{B}$, based on the aggregation of the deterioration state estimates of the structural categories $C_{1:\mathtt{K}}$, with the aggregated inspections $\tilde{y}^b_t \in [25, 100]$ represented by the blue diamond, and their corresponding uncertainty estimates represented by the blue error bars. The inspections represented by a magenta square correspond to synthetic inspections on a trajectory of deterioration that is generated based on the RL agent interventions, which are suggested at years 2022 and 2029.

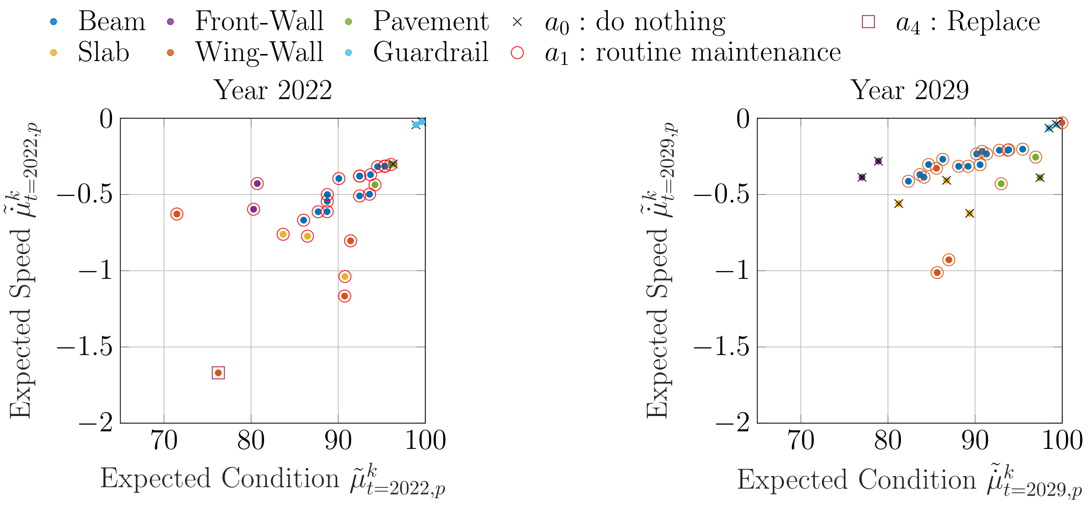


Fig. 12. Scatter plot for the expected deterioration condition versus the expected deterioration speed for all elements of bridge $\mathcal{B}$, with maintenance actions suggested by the HRL agent at the years 2022 (on the left) and year 2029 (on the right).

for each structural category. The other limitation in the proposed HRL is the use of a bottom-to-top approach for learning the policies with fixed policies π_k [36]. Alleviating these limitation could be done by using the policies π_k as a source policy that provide demonstrations for an end-to-end hierarchical RL training.

5. Conclusion

This paper introduce a hierarchical formulation and a RL environment for planning maintenance activities on bridges. The proposed formulation enables decomposing the bridge maintenance task into sub-tasks by using a hierarchy of policies, learned via deep reinforcement learning. In addition, the hierarchical formulation incorporates the deterioration speed in the decision-making analyses by relying on a SSM-based deterioration model for estimating the structural deterioration over time. A case study of a bridge is considered to demonstrate the applicability of the proposed approach which is done in two parts. The first part considered varying the number of structural elements to examine the scalability of the proposed framework against existing deep RL frameworks, such as the branching dueling Q-network (BDQN) and multi agent advantage actor–critic (MAA2C). The results of the comparison have shown that the proposed hierarchical approach has a better scalability than BDQN and MAA2C while sustaining a similar performance in cases with small number of structural elements. The second part in the case study addressed a maintenance planning problem for a bridge with multiple structural categories. In this case, the HRL agent performance is demonstrated by the element-level maintenance actions performed over a span of 10 years.

Overall, this study has demonstrated the capacity to learn a maintenance policy using hierarchical RL for a bridge with multiple structural categories. In addition, the analyses have highlighted the role of the deterioration speed in the decision-making process. Further extensions to this framework may include a multi-agent setup to learn a network-level maintenance policies under budgetary constraints, as well as designing and testing RL frameworks that can handle the uncertainty associated with the deterioration state in a POMDP environment. The contributions in this paper also include an open-source RL benchmark environment (link: https://github.com/CivML-PolyMtl/InfrastructuresPlanner), which is made available for contributions by the research community. This RL environment provides a common ground for designing and developing maintenance planning policies, in addition to comparing different maintenance strategies.

CRediT authorship contribution statement

Zachary Hamida: Writing – review & editing, Writing – original draft, Visualization, Validation, Methodology, Investigation, Formal

analysis, Data curation, Conceptualization. **James-A. Goulet:** Writing – review & editing, Supervision, Resources, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The experiments in this research work are performed using an open-source RL environment named InfraPlanner, which can be accessed at (link: https://github.com/CivML-PolyMtl/InfrastructuresPlanner).

Acknowledgments

This project is funded by the Transportation Ministry of Quebec Province (MTQ), Canada. The authors would like to acknowledge the support of Simon Pedneault for facilitating the access to information related to this project.

Appendix A. Deep reinforcement learning

A.1. Dueling deep network

In the context where a state s has similar $Q(s,a;\boldsymbol{\theta})$ values for different actions $a \in \mathcal{A}$, learning the value function $v(s)$ for each state can facilitate in learning the optimal policy π^*. The dueling network architecture enables incorporating the value function $v(s)$ in the Q-learning by considering,

$$Q(s_t,a_t;\boldsymbol{\theta}_\alpha,\boldsymbol{\theta}_\beta) = V(s_t;\boldsymbol{\theta}_\alpha) + adv(s_t,a_t;\boldsymbol{\theta}_\beta), \tag{A.1}$$

where $adv(s,a)$ is the approximation for the advantage of taking action a in state s, and $\boldsymbol{\theta}_\alpha, \boldsymbol{\theta}_\beta$ are the set of parameters associated with value function and the advantage function, respectively. Further details about the dueling network architecture are available in the work of Wang et al. [37].

A.2. Multi agent advantage actor–critic (MAA2C)

MAA2C is a multi agent extension of the advantage actor–critic A2C algorithm, where the centralized critic learns a joint state value from all the agents [22]. In this context, each actor loss is defined by,

$$\mathcal{L}_{a(i)}(\boldsymbol{\theta}_i) = -\log\pi(a_t|s_t;\boldsymbol{\theta}_i)\left(r_t + \gamma\, V(s_{t+1};\boldsymbol{\theta}_\alpha) - V(s_t;\boldsymbol{\theta}_\alpha)\right), \tag{A.2}$$

while the critic loss $\mathcal{L}_c$ is defined by the mean squared error as in,

$$\mathcal{L}_c(\boldsymbol{\theta}_\alpha) = \mathbb{E}\left[r_t + \gamma\, V(s_{t+1};\boldsymbol{\theta}_\alpha) - V(s_t;\boldsymbol{\theta}_\alpha)\right]^2. \tag{A.3}$$

A.3. DRL hyperparameters

The RL agents at all levels are trained using a discount factor 0.99, while relying on a batch size of 50 samples. The environment is vectorized to accelerate the training process and improve the sample independence, as the agent simultaneously interacts with $\mathtt{n} = 50$ randomly seeded environments. The exploration is performed using $\epsilon-$greedy, which is annealed linearly over the first 200 episodes with minimum $\epsilon_{\min} = 0.01$. Furthermore, the target model updates are performed every 100 steps in the environment. All neural networks have the same architecture (for the structural categories and the bridge) which consists in 2 layers of 128 hidden units and $relu(\cdot)$ activation functions. The learning rate for the off-policy agents starts at 10^{-3} and is reduced to 10^{-5} after 800 episodes, while for the on-policy agents the learning rate starts at 10^{-4}. The choice of hyperparameters for the RL agents is determined by using a grid-search for different combinations of values.

Appendix B. Environment configuration

This section presents some of the predefined functions in the environment which are based on previous work and numerical experiments.

B.1. SSM-based deterioration model

The Kalman filter (KF) describes the transition over time t, from the hidden state $\boldsymbol{x}_{t-1}$ to the hidden state $\boldsymbol{x}_t$ using the prediction step and the update step. The prediction step is described by,

$$\begin{aligned}\mathbb{E}[\boldsymbol{X}_t|\boldsymbol{y}_{1:t-1}] &\equiv \boldsymbol{\mu}_{t|t-1} = \boldsymbol{A}_t\boldsymbol{\mu}_{t-1|t-1}\\ \operatorname{cov}[\boldsymbol{X}_t|\boldsymbol{y}_{1:t-1}] &\equiv \boldsymbol{\Sigma}_{t|t-1} = \boldsymbol{A}_t\boldsymbol{\Sigma}_{t-1|t-1}\boldsymbol{A}^\intercal + \boldsymbol{Q}_t,\end{aligned}$$

where $\mathbb{E}[\boldsymbol{X}_t|\boldsymbol{y}_{1:t-1}]$ the expected value and $\operatorname{cov}[\boldsymbol{X}_t|\boldsymbol{y}_{1:t-1}]$ represent the covariance associated with the hidden state vector $\boldsymbol{x}_t$ given all the observations $\boldsymbol{y}_{1:t-1}$ up to time $t-1$, $\boldsymbol{A}_t$ is the transition matrix and $\boldsymbol{Q}_t$ is the model process-error covariance. In this context, the transition matrix $\boldsymbol{A}_t$ is time dependent such that,

$$\boldsymbol{A}_{t=\tau} = \begin{bmatrix} \boldsymbol{A}^{\text{ki}} & \boldsymbol{I}_{3\times3}\\ \boldsymbol{0}_{3\times3} & \boldsymbol{I}_{3\times3}\end{bmatrix},\ \boldsymbol{A}_{t\neq\tau} = \begin{bmatrix} \boldsymbol{A}^{\text{ki}} & \boldsymbol{0}_{3\times3}\\ \boldsymbol{0}_{3\times3} & \boldsymbol{I}_{3\times3}\end{bmatrix},\ \boldsymbol{A}^{\text{ki}} = \begin{bmatrix} 1 & dt & \frac{dt^2}{2}\\ 0 & 1 & dt\\ 0 & 0 & 1\end{bmatrix},$$

where τ is the time of the element-level maintenance action, and $\boldsymbol{I}$ is the identity matrix. Accordingly, the covariance matrix $\boldsymbol{Q}_t$ is described by,

$$\boldsymbol{Q}_{t=\tau} = \begin{bmatrix} \boldsymbol{Q}^{\text{ki}} + \boldsymbol{Q}^r & \boldsymbol{0}_{3\times3}\\ \boldsymbol{0}_{3\times3} & \boldsymbol{Q}^r\end{bmatrix},\ \boldsymbol{Q}_{t\neq\tau} = \begin{bmatrix} \boldsymbol{Q}^{\text{ki}} & \boldsymbol{0}_{3\times3}\\ \boldsymbol{0}_{3\times3} & \boldsymbol{0}_{3\times3}\end{bmatrix},$$

with $\boldsymbol{Q}^r$ and $\boldsymbol{Q}^{\text{ki}}$ defined as,

$$\boldsymbol{Q}^r = \operatorname{diag}\left(\left[\sigma_{w_r}^2\ \dot{\sigma}_{w_r}^2\ \ddot{\sigma}_{w_r}^2\right]\right),\ \boldsymbol{Q}^{\text{ki}} = \sigma_w^2\begin{bmatrix}\frac{dt^5}{20} & \frac{dt^4}{8} & \frac{dt^3}{6}\\ \frac{dt^4}{8} & \frac{dt^3}{3} & \frac{dt^2}{2}\\ \frac{dt^3}{6} & \frac{dt^2}{2} & dt\end{bmatrix},$$

where dt is the time step size, σ_w is a model parameter that describes the process noise and $\boldsymbol{Q}^r$ is a diagonal matrix containing model parameters associated with the element-level intervention errors [9]. Following the prediction step, if an observation is available at any time t, the expected value and covariance are updated with the observation using the update step,

$$\begin{aligned} f(\boldsymbol{x}_t|\boldsymbol{y}_{1:t}) &= \mathcal{N}(\boldsymbol{x}_t;\boldsymbol{\mu}_{t|t},\boldsymbol{\Sigma}_{t|t})\\ \boldsymbol{\mu}_{t|t} &= \boldsymbol{\mu}_{t|t-1} + \boldsymbol{K}_t(\boldsymbol{y}_t - \boldsymbol{C}\boldsymbol{\mu}_{t|t-1})\\ \boldsymbol{\Sigma}_{t|t} &= (\boldsymbol{I} - \boldsymbol{K}_t\boldsymbol{C})\boldsymbol{\Sigma}_{t-1|t-1}\\ \boldsymbol{K}_t &= \boldsymbol{\Sigma}_{t-1|t-1}\boldsymbol{C}^\intercal\boldsymbol{G}_t^{-1}\\ \boldsymbol{G}_t &= \boldsymbol{C}\boldsymbol{\Sigma}_{t-1|t-1}\boldsymbol{C}^\intercal + \boldsymbol{\Sigma}_V,\end{aligned}$$

where $\boldsymbol{\mu}_{t|t} \equiv \mathbb{E}[\boldsymbol{X}_t|\boldsymbol{y}_{1:t}]$ is the posterior expected value and $\boldsymbol{\Sigma}_{t|t} \equiv \operatorname{cov}[\boldsymbol{X}_t|\boldsymbol{y}_{1:t}]$ represents the covariance, conditional to observations up to time t, $\boldsymbol{K}_t$ is the Kalman gain, and $\boldsymbol{G}_t$ is the innovation covariance. The monotonicity throughout the estimation process is imposed by relying on the deterioration speed constraints: $\dot{\mu}_{t|t} + 2\sigma_{t|t}^{\dot{x}} \leq 0$; which are examined at each time step t, and enacted using the PDF truncation method [38].

Aggregating the deterioration states is performed using a Gaussian mixture reduction (GMR) [39], which is employed to approximate a PDF of E_m Gaussian densities into a single Gaussian PDF by using,

$$\begin{aligned}\boldsymbol{\mu}_{t|\mathrm{T},m}^{j,*} &= \sum_{p=1}^{\mathrm{E}_m}\lambda_p^j\boldsymbol{\mu}_{t|\mathrm{T},p}^j,\\ \boldsymbol{\Sigma}_{t|\mathrm{T},m}^{j,*} &= \sum_{p=1}^{\mathrm{E}_m}\lambda_p^j\boldsymbol{\Sigma}_{t|\mathrm{T},p}^j + \sum_{p=1}^{\mathrm{E}_m}\lambda_p^j(\boldsymbol{\mu}_{t|\mathrm{T},p}^j - \boldsymbol{\mu}_{t|\mathrm{T},m}^{j,*})(\boldsymbol{\mu}_{t|\mathrm{T},p}^j - \boldsymbol{\mu}_{t|\mathrm{T},m}^{j,*})^\intercal,\end{aligned}$$

where $\boldsymbol{\mu}_{t|\mathrm{T},m}^{j,*}$ is the aggregated expected value, and λ_p^j is the weight associated with the contribution of the deterioration state of the structural element. The merging of the E_m Gaussian densities is moment-preserving, where the total covariance $\boldsymbol{\Sigma}_{t|\mathrm{T},m}^{j,*}$ consists in the summation

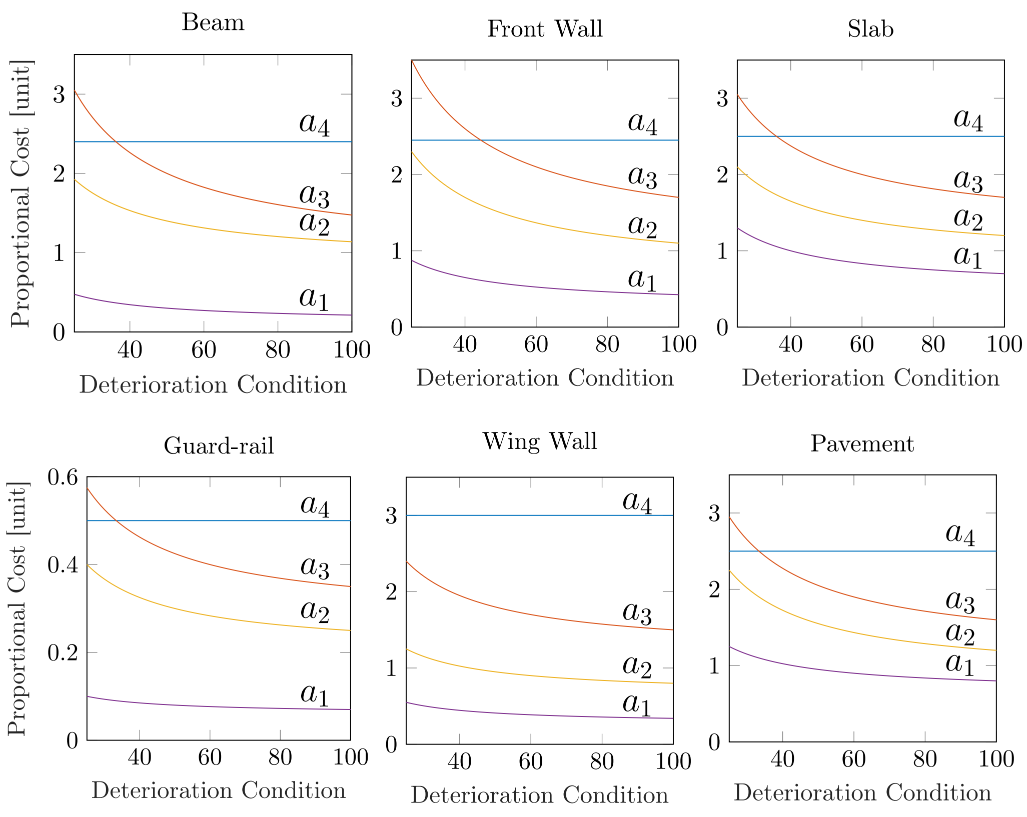


Fig. B.13. The proportional cost of each element-level action as a function of the deterioration condition.

of the "within-elements" contribution to the total variance, and the "between-elements" contribution to the total variance [8,39].

B.2. Decaying factors for effect of interventions

Decaying factors are introduced to prevent having the same improvement effect on structural elements while applying the same action within a short a period of time. The decaying factors in this context rely on the estimate for the expected time (number of years) to return to the state prior to the intervention [9]. Accordingly, the effect of intervention for any element-level action a^e, at time t is,

$$\delta^e = \rho_1 \times \delta^e,$$

where α_1 is the decaying factor defined as, $\rho_1 \propto \Pr(X_{\tau+t} \leq x_{\tau-1}|a)$, and τ is the time of intervention.

B.3. Maintenance actions effects & costs

Maintenance actions at the element level have different effects on the structural health condition, mainly depending on the structural category type. The deterministic maintenance effects associated with each action are defined in Table B.1. It should be noted that the values defined in Table B.1 have been derived from estimates that are based on data from the network of bridges in the province of Quebec [9].

As for the cost of maintenance actions, the cost functions are considered to be dependent on the deterioration state using,

$$x_c(\tilde{x}^k_{t,p}, a) = \beta_1(a)\frac{1}{\tilde{x}^k_{t,p}} + \beta_2(a),$$

where $\beta_1(a)$ is the cost of performing the maintenance action a as a function of the deterioration state $x^k_{t,p}$, and $\beta_2(a)$ is a fixed cost associated with maintenance action a. The derivation of this relation is empirical and mimics the cost information provided by the ministry of transportation in Quebec. Fig. B.13 shows the proportional cost function for the elements within each structural category. From the graphs in Fig. B.13, it is noticeable that the replacement cost is considered fixed and independent of the structural condition. Moreover, in some cases, the cost of performing an action may exceed the cost of replacement.

Based on the cost function $x_c(\cdot)$, the element-level rewards $r(s^e_{t,p}, a^k_{t,p})$ are defined as,

$$r(s^e_{t,p}, a^k_{t,p}) = x_c(\tilde{x}^k_{t,p}, a^k_{t,p}) + r^p,$$

where r^p is the penalty applied when a predefined critical state is reached and no maintenance action is taken.

Table B.1
Table for the true improvement on the health condition based on the element-level maintenance actions and each structural category.

Action	Structural Category					
	Beams	Front wall	Slabs	Guardrail	Wing wall	Pavement
a_0	0	0	0	0	0	0
a_1	0.5	0.1	1	0.25	0.25	8
a_2	7.5	19	12	9	8	20
a_3	18.75	20.5	20	14	17	28
a_4	75	75	75	75	75	75

References

[1] Asghari Vahid, Biglari Ava Jahan, Hsu Shu-Chien. Multiagent reinforcement learning for project-level intervention planning under multiple uncertainties. J Manage Eng 2023;39.
[2] Andriotis Charalampos P, Papakonstantinou Konstantinos G. Managing engineering systems with large state and action spaces through deep reinforcement learning. Reliab Eng Syst Saf 2019;191:106483.
[3] Wei Shiyin, Bao Yuequan, Li Hui. Optimal policy for structure maintenance: A deep reinforcement learning framework. Struct Saf 2020;83.
[4] Nguyen Van Thai, Do Phuc, Vosin Alexandre, Iung Benoit. Artificial-intelligence-based maintenance decision-making and optimization for multi-state component systems. Reliab Eng Syst Saf 2022;228.

[5] Hamida Zachary, Goulet James-A. Modeling infrastructure degradation from visual inspections using network-scale state-space models. Struct Control Health Monit 2020;1545–2255.

[6] Moore Mark, Phares Brent M, Graybeal Benjamin, Rolander Dennis, Washer Glenn. Reliability of visual inspection for highway bridges, volume I. Technical report, Turner-Fairbank Highway Research Center; 2001.

[7] Agdas Duzgun, Rice Jennifer A, Martinez Justin R, Lasa Ivan R. Comparison of visual inspection and structural-health monitoring as bridge condition assessment methods. J Perform Constr Facil 2015;30(3):04015049.

[8] Hamida Zachary, Goulet James-A. A stochastic model for estimating the network-scale deterioration and effect of interventions on bridges. Struct Control Health Monit 2021;1545–2255.

[9] Hamida Zachary, Goulet James-A. Quantifying the effects of interventions based on visual inspections of bridges network. Struct Infrastruct Eng 2021;1–12. http://dx.doi.org/10.1080/15732479.2021.1919149.

[10] MTQ. Manuel d'inspection des structures. Ministère des Transports, de la Mobilité Durable et de l'Électrification des Transports; 2014.

[11] Du Ao, Ghavidel Alireza. Parameterized deep reinforcement learning-enabled maintenance decision-support and life-cycle risk assessment for highway bridge portfolios. Struct Saf 2022;97.

[12] Lei Xiaoming, Xia Ye, Deng Lu, Sun Limin. A deep reinforcement learning framework for life-cycle maintenance planning of regional deteriorating bridges using inspection data. Struct Multidiscip Optim 2022;65.

[13] Sutton Richard S, Barto Andrew G. Reinforcement learning: an introduction. MIT Press; 2018.

[14] Fereshtehnejad Ehsan, Shafieezadeh Abdollah. A randomized point-based value iteration POMDP enhanced with a counting process technique for optimal management of multi-state multi-element systems. Struct Saf 2017;65:113–25.

[15] Yang David Y, Asce AM. Deep reinforcement learning-enabled bridge management considering asset and network risks. J Infrastruct Syst 2022;28(3):04022023.

[16] Zhang Nailong, Si Wujun. Deep reinforcement learning for condition-based maintenance planning of multi-component systems under dependent competing risks. Reliab Eng Syst Saf 2020;203.

[17] Zhou Yifan, Li Bangcheng, Lin Tian Ran. Maintenance optimisation of multicomponent systems using hierarchical coordinated reinforcement learning. Reliab Eng Syst Saf 2022;217.

[18] Kok Jelle R, Vlassis Nikos. Collaborative multiagent reinforcement learning by payoff propagation. J Mach Learn Res 2006;7:1789–828.

[19] Abdoos Monireh, Mozayani Nasser, Bazzan Ana LC. Holonic multi-agent system for traffic signals control. Eng Appl Artif Intell 2013;26(5–6):1575–87.

[20] Jin Junchen, Ma Xiaoliang. Hierarchical multi-agent control of traffic lights based on collective learning. Eng Appl Artif Intell 2018;68:236–48.

[21] Tavakoli Arash, Pardo Fabio, Kormushev Petar. Action branching architectures for deep reinforcement learning. In: Proceedings of the AAAI conference on artificial intelligence. 2018.

[22] Papoudakis Georgios, Christianos Filippos, Schäfer Lukas, Albrecht Stefano V. Benchmarking multi-agent deep reinforcement learning algorithms in cooperative tasks. In: Conference on neural information processing systems track on datasets and benchmarks. 2021.

[23] Kuba Jakub Grudzien, Wen Muning, Meng Linghui, Zhang Haifeng, Mguni David, Wang Jun, et al. Settling the variance of multi-agent policy gradients. Adv Neural Inf Process Syst 2021;34:13458–70.

[24] Abel David. A theory of state abstraction for reinforcement learning. In: Proceedings of the AAAI conference on artificial intelligence. 2019.

[25] Abel David, Hershkowitz David, Littman Michael. Near optimal behavior via approximate state abstraction. In: International conference on machine learning. PMLR; 2016, p. 2915–23.

[26] Brockman Greg, Cheung Vicki, Pettersson Ludwig, Schneider Jonas, Schulman John, Tang Jie, et al. OpenAI gym. 2016, Arxiv.

[27] Hamida Zachary, Goulet James-A. Network-scale deterioration modelling based on visual inspections and structural attributes. Struct Saf 2020;88:102024.

[28] Pateria Shubham, Subagdja Budhitama, Tan Ah Hwee, Quek Chai. Hierarchical reinforcement learning: A comprehensive survey. ACM Comput Surv 2021;54.

[29] Watkins Christopher John Cornish Hellaby. Learning from delayed rewards [Ph.D. thesis], King's College, Cambridge United Kingdom, University of Cambridge; 1989.

[30] Kobayashi Taisuke, Ilboudo Wendyam Eric Lionel. T-soft update of target network for deep reinforcement learning. Neural Netw 2021;136:63–71.

[31] Nachum Ofir, Gu Shixiang Shane, Lee Honglak, Levine Sergey. Data-efficient hierarchical reinforcement learning. Adv Neural Inf Process Syst 2018;31.

[32] Gronauer Sven, Diepold Klaus. Multi-agent deep reinforcement learning: A survey. Artif Intell Rev 2022;55:895–943.

[33] Kalman Rudolph Emil. A new approach to linear filtering and prediction problems. J Basic Eng 1960;82(1):35–45.

[34] Kanervisto Anssi, Scheller Christian, Hautamäki Ville. Action space shaping in deep reinforcement learning. In: 2020 IEEE conference on games. IEEE; 2020, p. 479–86.

[35] Zhu Zhuangdi, Lin Kaixiang, Zhou Jiayu. Transfer learning in deep reinforcement learning: A survey. 2020, ArXiv.

[36] Florensa Carlos, Duan Yan, Abbeel Pieter. Stochastic neural networks for hierarchical reinforcement learning. 2017, Arxiv.

[37] Wang Ziyu, Schaul Tom, Hessel Matteo, Hasselt Hado, Lanctot Marc, Freitas Nando. Dueling network architectures for deep reinforcement learning. In: International conference on machine learning. PMLR; 2016, p. 1995–2003.

[38] Simon Dan, Simon Donald L. Constrained Kalman filtering via density function truncation for turbofan engine health estimation. Internat J Systems Sci 2010;41(2):159–71.

[39] Runnalls Andrew R. Kullback-Leibler approach to Gaussian mixture reduction. IEEE Trans Aerosp Electron Syst 2007;43(3):989–99.