Universiteit Leiden
The Netherlands

Model-assisted optimal control framework for industrial system coupling problems

Wei, X.

Citation

Wei, X. (2026, July 7). *Model-assisted optimal control framework for industrial system coupling problems*. Retrieved from https://hdl.handle.net/1887/4307775

Version:	Publisher's Version
License:	Licence agreement concerning inclusion of doctoral thesis in the Institutional Repository of the University of Leiden
Downloaded from:	https://hdl.handle.net/1887/4307775

Note: To cite this publication please use the final published version (if applicable).

CHAPTER 3

Case Study on Optimal Control Problems under System–Environment Coupling Conditions

This chapter takes the trajectory navigation problem of a tunnel boring machine (TBM) in underground environments as a case study to explore optimal control methods under equipment–environment coupling conditions. In this study, we propose a hierarchical POMDP framework that leverages sequences of observable states to perform belief inference about unobservable states, thereby addressing control decision-making in partially observable environments.

3.1 Background

TBMs are a type of large engineering machine that have been widely used in large-scale infrastructure projects including metro systems, railway and highway tunnels, water conveyance pipelines, and urban utility tunnels (Li et al., 2024; Mahmoodzadeh et al.; 2022; Zhang et al., 2023). TBMs have received the most attention in terms of precise control to ensure tunneling quality and excavation safety (Li et al., 2024). Typically, the allowable deviation for TBMs excavation path is limited to ± 50 mm. As the actual excavation path is the joint output of the guidance of designed trajectory and real-time manipulation, it is essential to meticulously track and correct the trajectory with appropriate control parameters during the operation of TBMs.

The factors determining the excavation trajectory are categorized as external or internal. The main external factor is the geotechnical environment. Current detection methods cannot accurately and comprehensively capture geotechnical information along the excavation paths. (Li et al., 2017; Pan et al., 2023; Chen et al., 2017; Huang et al., 2021) Geotechnical exploration before construction using interval drilling and sampling can only provide some preliminary stratigraphical profiles (Abdollahi et al., 2019; Adoko et al., 2019) . This exploration is very limited compared with the tunnel length, leaving the TBM operating in an uncertain environment and causing time-varying and unpredictable disturbances. Internal factors include the complex structure and driving force of TBM. First, the motion of TBM is the result of interactions among various components, such as the shield body, cutterhead, driving system, hydraulic thrust system, and rear trolley (Huang ta al., 2021). Consequently, the excavation process is restricted by conditions such as the cutter type, driving system torque, hydraulic thrust capacity, and overall weight. Second, during excavation, multiple forces, such as kinetic energy, frictional forces, resistance, and electromagnetic energy, are involved. Owing to lateral soil and rock resistance, TBM can only move perpendicular to the cutterhead. This creates a non-holonomic constrained motion that complicates the computation of optimal TBM control parameters through analytical methods (Sugimoto et al., 2007). Therefore, trajectory deviations and empirical corrections are routine during excavation.

TBM trajectory navigation involves adjusting the control parameters at each time step to minimize the error between the actual path and the designed trajectory. In this process, the thrust force of the TBM serves as the direct control variable. Only by applying an appropriate thrust force can the TBM accurately follow the designed excavation trajectory. The thrust must balance the geotechnical loads at the excavation face. If the thrust is too low, the cutterhead may fail to effectively fracture the rock, resulting in excavation difficulties. Conversely, if the thrust is too high, it may disturb the surrounding strata, potentially causing ground uplift or collapse. A well-regulated thrust contributes to stabilizing the soil ahead of the TBM, thereby preventing issues such as water ingress, sand inflow, or cavity formation. However, due to the influence of surrounding geotechnical conditions, accurately computing the required thrust remains a persistent challenge.

To effectively control TBMs' thrust force despite partially observable information, a way that makes the best use of the available data is needed. Therefore, our goal is to build an end-to-end control resolution that converts the state data from a monitoring system, which includes the TBM's current position and orientation, as well as the design trajectory, along with the preliminary geotechnical data obtained from pre-construction surveys, into the optimal thrust force. It is worth noting that, in TBM trajectory navigation, there exists a causal relationship between unobservable information, i.e., geotechnical properties, and the observable motion states of the TBM. Specifically, the geotechnical environment surrounding the TBM is determined by its current position, which itself is defined by its historical trajectory. Furthermore, the surrounding geotechnical conditions tend to change gradually as the TBM advances along its path. This spatiotemporal continuity provides a promising opportunity to infer the required thrust over a future time horizon based on historical observable sequence during the excavation process.

Motivated by this insight, a hierarchical POMDP framework to suppress the impacts of partially observable information is proposed. The framework consists of three layers: a policy layer to make preliminary decisions utilizing observable information, a belief layer to inferring unobservable state distributions through observable state sequences, and a joint decision layer to achieve final decision-making. By decomposes POMDP into three layers, the proposed framework restructures the computational problem by isolating and handling different complexity sources separately, making it particularly suitable for real-world applications.

For applying the framework, the original problem of '*determining the thrust required to balance the geotechnical loads, such that the TBM can excavate along the predefined design trajectory*' is transformed to two tasks:

(1) TBM advance path planning. The goal is to determine the most feasible path that minimized the distance between the actual position of the TBM and the designed trajectory. By completing this task, an optimal excavation path time series can be obtained, which serves as the foundation for thrust prediction.

(2) TBM thrust force prediction. The target is to forecast the force conditions of the TBM to ensure that the output thrust force of the propulsion system maintaining consistency with the surrounding geotechnical loads. This task constitutes the ultimate objective of TBM trajectory navigation.

Policy layer aims to achieve successful path planning - corresponding to task (1) - which serves as a fundamental prerequisite for effective trajectory navigation. Given the sequential decision-making nature of this problem, we employ a Reinforcement Learning (RL) agent to determine the optimal excavation path. However, the operational conditions demonstrated by the available real-world TBM data are limited, rendering them incomplete for RL training. To address this issue, a Patch-based Contact-Driven Motion (PCDM) model of a TBM is developed for online RL training.

The belief and joint decision layers are designed to address thrust prediction, corresponding to Task (2). To leverage the temporal information of TBM operational states, a Gated Recurrent Unit (GRU) is employed to construct the belief layer, which infers the distribution of unobservable states from sequential TBM state data along the planned path. The inferred distribution is subsequently processed by a Deep Neural Network (DNN) in the joint decision layer to generate final thrust predictions. Furthermore, the generalization ability of the GRU and DNN model for force prediction is verified using tunning strategies such as transfer learning.

Based on the hierarchical POMDP framework, to achieve an end (measurements from the environment)-to-end (control parameters) solution for TBM, we propose a TBM trajectory navigation approach that consists of the following three components.

1. An RL agent that determines the optimal excavation path for the TBM and controls its movement.

2. A GRU model that predicts TBM forces based on historical TBM motion state sequence and inadequate geotechnical information.

3. A PCDM model of TBM that simulates its motion and force mechanism in a geotechnical environment.

3.2 Previous Research and Main Issue

The navigation of TBMs in uncertain subsurface environments presents a significant challenge, primarily due to the partially observable and spatially variable geotechnical conditions (Chen et al. 2023; Shao et al. 2019). Therefore, researchers in this field have primarily focused on enhancing TBM performance and ensuring excavation success through the development and application of advanced computational methods. In recent years, this has led to the growing application of digital twin, deep learning, and multi-objective optimization for ensuring the safe and efficient operation of TBMs underground while optimizing their control parameters (Zhang et al. 2024; Liu et al. 2022; Naghadehi et al. 2018). While existing research has contributed to ensuring long-term TBM operation in underground environments and reducing accidents and errors, previous studies have failed to address the fundamental challenge of TBM operation in uncertain conditions: unobservable geotechnical states can significantly compromise the accuracy of control decisions. Although attempts have been made to develop geological prediction methods, these have not been systematically integrated with TBM navigation and control (Mostafa et al. 2024, Fu et al. 2023, Zhang et al. 2023). To bridge this gap, a novel POMDP framework to address TBM navigation in unobservable subsurface environments is expected.

When designing a POMDP framework for TBM trajectory navigation, the primary challenge lies in effectively inferring unobservable subsurface geotechnical conditions. This necessitates a methodology capable of leveraging available observable information to reconstruct the belief state of unobservable state, which is critical for accurate decision-making. Crucially, throughout the TBM's operational timeline, the sequence of observable trajectory information - including thrust forces and kinematic paths - fundamentally encodes the dynamic changes of the surrounding geotechnical properties. However, prevailing approaches to POMDP problems have largely failed to fully exploit this temporal dependency. They often focus on instantaneous state estimations or myopic policies, overlooking the possible temporal causality between observable actions and unobservable state transitions (Zhang et al. 2025, Pajarinen et al. 2023, Song et al. 2022).

Moreover, in previous studies, researchers have typically adopted a two-step approach in which the unobservable states are first inferred and then concatenated with the observable states as to form an augmented input for downstream decision-making (Singla et al., 2021; Miki et al., 2022). However, this approach introduces inference errors at the initial stage of the decision-making process, thereby amplifying overall decision inaccuracies. To fully leverage this temporal causality, we propose a three-layer hierarchical POMDP framework comprising a policy layer, belief layer, and joint decision layer. By decomposing the overall task into specialized layers, compared to existing methods, this hierarchical decision-making architecture not only shortens error propagation paths and effectively reduces overall problem complexity, but also enables more effective utilization of temporal patterns embedded in the

TBM's operational state sequences for inference.

The objective of policy layer is to achieve successful path planning by minimizing the TBM's positional and angular deviations from the designed trajectory. Similar to many other decision-making problems, the complexity of TBM path planning increases exponentially with the number of steps. Consequently, determining the optimal solution through an exhaustive search, even with only a few hundred steps, is intractable. In this scenario, a learning method that is capable of addressing sequential decision problems in a large search space is required. Reinforcement Learning (RL), which mimics the nature of human learning, is an attractive alternative for solving the TBM path planning problem because it is state-of-the-art in addressing Markov decision process (MDP) problems (Lei et al., 2018; Hu et al., 2023). RL has been proven to be a powerful solution for reward-instructed autonomous control problems using a trial-and-error mechanism for continuous learning（Ye et al., 2022; Zhao et al., 2023; Perrusquía et al., 2022; Hao et al., 2024; Pan et al., 2024; Eskandari et al., 2024). In recent years, breakthroughs made by researchers in new RL algorithms (Silver et al., 2016; Yan et al., 2023) have driven RL to become faster and more stable (Haarnoja et al., 2018; Hessel et al., 2018).

In the belief layer and the joint decision layer, the primary challenge lies in devising methods that leverage partially observable geotechnical information to acquire the precise force conditions. Since TBM movement is inherently a temporal process, there exists a correlation between the current and past states. This implies that it is feasible to predict future thrust by utilizing the historical geotechnical loads sequence observed during excavation along the optimal path determined by the policy layer. The Gated Recurrent Unit (GRU) (Weerakody et al., 2021) has gained recognition for its effectiveness in time-series prediction and has emerged as a competitive solution. GRUs have been successfully applied in a wide range of domains, including natural language processing (Yu et al., 2024), speech recognition (Ravanelli et al., 2018), and sequence generation (Encalada-Davila et al., 2022; Liu et al., 2023). However, to the best of our knowledge, GRUs have not yet been applied to TBM operational state prediction. To make full use of the available information, preliminary geotechnical exploration data obtained before construction is also incorporated.

3.3 Hierarchical POMDP Framework

To tackle the TBM trajectory navigation challenge under partial observability, we construct a hierarchical POMDP framework aimed at improving the accuracy of decision-making. The POMDP refers to a Markov process designed for sequential decision-making under uncertain environments. Its core characteristic lies in the decision-maker's inability to directly access the system's true state, necessitating the inference of actual conditions solely through incomplete and noisy observational information. The POMDP can be represented as a tuple $M\left(S,A,T,R,\Omega,O,\gamma\right)$, where S denotes the state space, A represents the action space, T represents the state transition probability, $R:S\times A\to\mathbb{R}$ is the reward function, Ω is the set of observations, O is the conditional observation probability, and γ is the discount factor. We assume the state space S is a vector space and each state vector $s_t\in S$ is represented by the product of an observable and an unobservable/hidden component.

$$s_t = s_t^o \times s_t^h \tag{3.1}$$

where s_t^o is the observable component, while s_t^h represents the unobservable component. In this setting, the observations space Ω is the collection of all such s_t^o. Typically, s_t^o and s_t^h are causally related, and hence, it is standard to assume a probabilistic relation between them: $O(s_t^o \mid s_t, a_{t-1})$ - the probability of observing s_t^o after transiting into state s_t with action a_{t-1}. The research is concerned with that predicting the hidden component given the observed ones. It is stated that the predictive distribution of the hidden component s_{t+1}^h depends on the history of observed components. Starting with the following marginalization:

$$P(s_{t+1}^o, s_{t+1}^h \mid s_t^o, a_t) = \int P(s_{t+1}^o, s_{t+1}^h, s_t^h \mid s_t^o, a_t) ds_t^h \tag{3.2}$$

It is obtained that

$$P(s_{t+1}^o, s_{t+1}^h \mid s_t^o, a_t) = \int P(s_{t+1}^o, s_{t+1}^h \mid s_t^o, s_t^h, a_t) P(s_t^h \mid s_t^o) ds_t^h \tag{3.3}$$

Note that $P(s_{t+1}^o, s_{t+1}^h \mid s_t^o, s_t^h, a_t)$ should be $T(s_{t+1}, s_t, a_t)$, which is the transition probability. From Eq. (3.3), the conditional distribution of s_{t+1}^h on s_{t+1}^o and action a_t is expressed as:

$$P(s_{t+1}^h \mid s_{t+1}^o, a_t) = \int \frac{T(s_{t+1}, s_t, a_t)P(s_t^h \mid s_t^o)ds_t^h}{O(s_{t+1}^o, s_t^o, a_t)} \tag{3.4}$$

Now, marginalizing out the action, it is deduced that:

$$P(s_{t+1}^h \mid s_{t+1}^o) = \iint \frac{T(s_{t+1}, s_t, a_t)P(s_t^h \mid s_t^o)P(a_t)ds_t^h da_t}{O(s_o^{t+1}, s_o^t, a_t)} \tag{3.5}$$

From the above expression, it is clear that the conditional distribution of s_{t+1}^h on s_{t+1}^o can be described by a function of the conditional distribution of s_t^h on s_t^o - a recursive relationship. Directly modelling Eq. (3.5) is challenging since the transition probability $T(s_{t+1}, s_t, a_t)$ and the observation distribution $O(s_o^{t+1}, s_o^t, a_t)$ are difficult to determine explicitly. Given the temporal dependency inherent in the relationship between s_t^h and s_t^o, this study adopts a time-series prediction to estimate the unobservable components, which predicts s_{t+1}^h based on an n-step time horizon of the observations into the past:

$$s_{t+1}^h = f_c(s_t^o, s_{t-1}^o, \ldots, s_{t-n}^o, \varepsilon) \tag{3.6}$$

For systems in which state transitions exhibit inertia, particularly those with long-term dependencies across time steps, the state tends to display strong temporal characteristics. In the POMDP, although the system states are only partially observable, their long-term causal effects are still implicitly accumulated and reflected in the observation sequence. This temporal dependency, combined with the underlying causal mapping, can be described as:

$$s_{h,t} = T(s_{t-1}^o, s_{t-2}^o, \ldots, s_{t-k}^o, \theta, \varepsilon_t) \approx f_t(s_{t-1}^o, s_{t-2}^o, \ldots, s_{t-k}^o, \varepsilon_t) \tag{3.7}$$

where f_t denotes the time series prediction function, θ represents uncertain influencing factors, ε_t is the noise term, and k is the length of historical dependency. θ primarily refers to the action a_t at each time step, which can be regarded as an additional perturbation to the state sequence. The generation of a_t is governed by the decision policy π, implying that the prediction function f_t is influenced by π. When π is a short-sighted policy, the temporal consistency of the action sequence $\boldsymbol{a}$ tends to be weak, thereby disrupting the long-term temporal dependency of the state sequence. Furthermore, if a_t exerts a significant influence on the state s_t, the predictive capability of f_t may also degrade.

When a far-sighted policy is adopted, the influence of a_t on s_t is relatively smooth. If a_t does not vary drastically within a short time window, the state sequence can still retain strong temporal coherence. This characteristic is well exemplified in the case of TBM navigation, which exhibits the following features:

The TBM advances through a continuous excavation process. During the currently unobservable state, i.e., the surrounding geotechnical conditions is largely determined by the preceding excavation states.

Steering operations tend to be slow due to spatial constraints within the tunnel, which impose a relatively large minimum turning radius. As a result, the control input applied at each time step has a cumulative gradual effect on the overall motion posture. Moreover, since the primary objective of navigation is to follow the designed trajectory, real-time variations in control inputs are typically limited.

For this type of problem, particularly when the task can be decomposed at the strategic level rather than treated as a purely reactive task, it is feasible to improve POMDPs through task hierarchization. Typically, the task can be divided into two stages: a high-level decision stage that leverages deterministic states and policies to produce a preliminary plan, and a final decision stage that utilizes the temporal patterns embedded in the observable state sequences to refine and execute the decision. In this study, the specific approach involves decomposing the original POMDP into three hierarchical levels:

1. Policy layer

The task of the policy layer is to extract the fully observable states from the original POMDP and perform preliminary decision-making. Specifically, actions a_t are generated from the observable state s_t^o according to a policy π. A deterministic optimal policy π_o is found by maximizing the cumulative reward under observable states. This process is formulated as:

$$\begin{gathered} a_t \sim \pi_o(s_t^o) \\ \max_{\pi_o} \mathbb{E}^{\pi_o}\left[\sum_{t=0}^{T} \gamma^t r_t \mid s_0^o\right] \end{gathered} \tag{3.8}$$

In the context of TBM trajectory navigation, the observable TBM position and orientation information is first utilized to perform underground path planning. It serves as a preliminary deterministic decision outcome, which in turn drives the subsequent decision-making process.

The task of the policy layer is referred as path planning, and its detailed formulation is discussed in Section 4.

2. Belief layer

The belief layer constructs a belief state estimator that leverages the temporal patterns in the sequence of observable states to infer the unobservable states. The objective of the inference process is to minimize the prediction error of the unobservable states. The process can be described as:

$$b_t(s_t^h), \hat{s}_t^h = f_c(s_{t-1}^o, s_{t-2}^o \cdots s_{t-n}^o) \tag{3.9}$$

where f_c denotes the temporal prediction function, $\hat{s}_t^h$ represents the estimated belief state, and b_t denotes the probability distribution over the belief state. In the context of TBM trajectory navigation, the sequence of TBM motion states obtained from the policy layer is used to generate a time series of geotechnical loads. By analyzing the temporal patterns in this sequence, the belief distribution $b_t(s_t^h)$ is inferred. Since there is no ground-truth value of the s_t^h available as training labels, the estimated belief state $\hat{s}_t^h$ is not explicitly obtained in this study. Instead, the belief distribution $b_t(s_t^h)$ is directly propagated to the joint decision layer for thrust prediction. This layer modeled the temporal information embedded in the observable state sequence to compensate for the influence of unobservable states. In essence, it utilizes the historical dynamic responses along the trajectory to implicitly encode the characteristics of the surrounding geotechnical environment.

3. Joint decision layer

The joint decision layer is to integrate the observable states s_t^o with the inferred unobservable states in order to generate the final action a_t^*. The decision function at this layer can be formulated as:

$$a_t^* = D(s_t^o, b_t(s_t^h)) \tag{3.10}$$

The objective is to maximize the cumulative reward under the full state, which can be expressed as:

$$\max_{D} \mathbb{E}^{D}\left[\sum_{t=0}^{T} \gamma_j^t r_t^j \mid s_0\right] \\ r_t^j = \mathcal{R}(a_t^*, s_t) \text{ or } r_t = \mathcal{R}(a_t^*, (s_t^o, b_t(s_t^h))) \tag{3.11}$$

where $\mathcal{R}$ denotes the reward function. The specific form of $\mathcal{R}$ should be determined by whether the unobservable state s_t^h can be explicitly expressed. Alternatively, the objective can be defined as minimizing the error between the final output and the ground truth, which can be formulated as:

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left\|a_i^* - a_i^{true}\right\|^2, \; a_t^* = f_{out}(a_t, s_t^h, \theta) \tag{3.12}$$

By integrating the three layers described above into a unified framework, the proposed framework provides a joint optimization solution to this type of POMDP problems. The overall objective can be formulated as a joint optimization problem:

$$\max_{D} \mathbb{E}^{D}\left[\sum_{t=0}^{T} \gamma_j^t r_t^j \mid s_0\right] \\ s.t. \begin{cases} s_t^o \sim T(s_{t-1}^o, a_t) \\ o_t \sim O(s_t^o, a_t, \delta_t) \\ b_t(s^h), \hat{s}_t^h = f_c(s_{t-1}^o, s_{t-2}^o \cdots s_{t-k}^o) \\ a_t^* = D(s_t^o, b_t(s_t^h)) \end{cases} \tag{3.13}$$

In the TBM trajectory navigation task, a Deep Neural Network (DNN) is employed to construct the joint decision layer, which outputs the predicted thrust for the TBM. The hierarchical POMDP framework and its application on TBM trajectory navigation is interpreted in Figure 3.1.

The hierarchical POMDP framework embodies a fundamental paradigm shift from replacing to augmenting expert knowledge, computationally formalizing engineers' cognitive decision-making through its layered architecture. A key breakthrough of this framework lies in its transformation of state estimation and decision-making from a serial pipeline to a parallel process. This architectural shift enables the system to actively explore and anticipate critical uncertainties while mitigating the error propagation and amplification inherent in traditional sequential decision-making. By integrating a contact dynamics-based simulation-to-reality approach, physical priors are embedded into model parameters, effectively addressing engineering data scarcity and providing a viable pathway for converting qualitative expert knowledge into computable constraints. It ensures deep integration of domain expertise while maintaining decision transparency through belief states, ultimately achieving trustworthy human-machine collaborative decision-making.

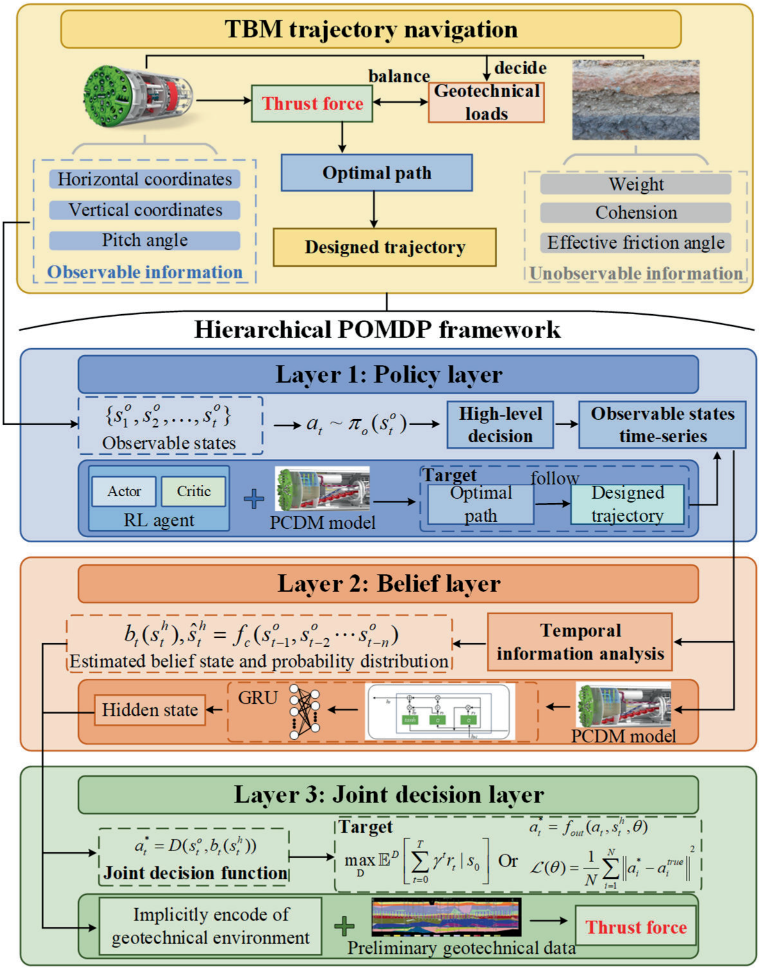


Figure 3.1: The hierarchical POMDP framework and its application on TBM trajectory navigation. The framework employs a three-layer hierarchical architecture: the policy layer generates initial decisions to accomplish successful path planning while producing observable state sequence; the belief layer processes temporal information to establish belief state distributions; finally, the joint decision layer synthesizes both known states and inferred unobservable states to determine optimal final decisions.

3.4 Methodology

This section formulates optimal path planning and thrust force prediction under the proposed hierarchical POMDP framework and presents the design of the TBM trajectory navigation approach. Furthermore, a transfer learning approach is discussed to generalize the distribution of a pre-trained thrust force prediction model to real-world scenarios.

3.4.1 Formulation of TBM Trajectory Navigation

TBM path planning

The objective of TBM path planning is to determine the shortest path that minimizes the deviation from the design trajectory while adhering to the minimum turning radius constraint. The movements of TBMs in the horizontal and vertical directions are similar. Therefore, this study just focused on the vertical direction of the motion process. The motion in the horizontal direction can be derived analogously. The vertical path planning process is depicted in Figure 3.2.

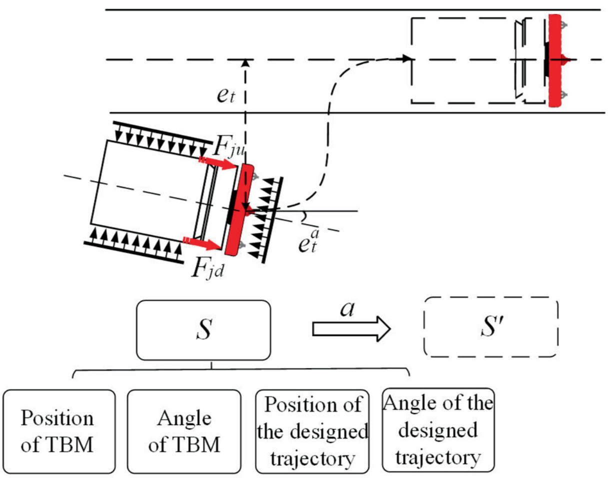


Figure 3.2: Process of TBM path planning.

The TBM advances at a stable rate, and the advancing direction is gradually changed to reduce the position error e_t and angular error e_t^a. We propose the following hypotheses to describe the movement of TBMs. 1) The excavation rate of TBMs is stable owing to the combined effect of the cutterhead rotation speed, penetration depth, and thrust force of the advancing system, which is also a requirement for practical construction. 2) The TBM control system has a refresh rate, and the TBM operates with fixed control parameters in one time step. Under these assumptions, the motion model of TBM focuses on TBM advances at a constant excavation rate and direction within a single time step.

Therefore, TBM path planning is considered as a sequential decision-making problem that can be formulated into a finite MDP with a discrete time step. This MDP is defined by a tuple $M\left(S^o, A, f, R, \gamma\right)$, where S^o represents the state space, A denotes the action space, f represents the state transition function, R is the reward function, and γ represents a discount factor. These five elements are used to describe TBM path planning as follows:

State space: Information about the TBM's position relative to the designed trajectory—which directly influence the control actions—should be included in the observation space. Therefore, the observation state is denoted as:

$$s_t^o = (\boldsymbol{P}_t, \theta_t, y_t^{df}, \theta_t^{df}) \tag{3.14}$$

where $\boldsymbol{P}_t = (x_t, y_t)$, x_t and y_t represent the horizontal and vertical positions of the TBM cutterhead center point at time step *t*, respectively, θ_t is the pitch angle of the TBM, and y_t^{df} and θ_t^{df} are vertical position and angle of the designed trajectory at a distance ε in the forward horizontal direction of the TBM, respectively. ε is defined as a preemptive distance of 5 m, specifically tailored to account for the fact that the TBM moves exclusively in the direction perpendicular to the cutterhead.

Action space: This study focuses on the TBM's adjustment of its motion direction within the vertical plane. Therefore, the pitch angle increment is considered as a continuous action a_t. The difference in movement distance between each part of the thrust system can be calculated using a_t and the TBM dimensional parameters.

State transaction: The transition function is defined as:

$$s_{t+1}^o = f(s_t^o, a_t) \tag{3.15}$$

The position and orientation will be updated according to a_t in the PCDM model.

Reward function: The immediate reward $r_t = R(s_t^o, a_t)$ is used to evaluate the error in the position and angle between the TBM and design trajectory at time t, which is the instantaneous feedback of a_t. In addition, the pitch angle of the TBM should be considered when evaluating the success of the correction. The reward function is defined as follows:

$$R(s_t^o, a_t) = \frac{2000}{(1 + e^{(5(\frac{e_t}{0.8} + \frac{e_t^a}{0.05}) - 2)})} - 1000 \tag{3.16}$$

The reward function is based on a sigmoid function and considers both the positional error e_t and the angular error e_t^a. As the error increases, the reward function decays rapidly towards its minimum value. Conversely, when the error approaches zero, the reward increases rapidly. During training, the value of the reward function is normalized by scaling to $[-1,1]$.

TBM thrust force prediction

Regarding TBM thrust force prediction, the primary aim is to accurately compute the thrust force required for TBM advancement, thereby achieving force equilibrium along the forward direction. In the dynamic load theory model proposed by Sugimoto (Sugimoto et al., 2002; Sramoon et al., 2002), the TBM is subjected to resistance from the soil ahead, peripheral friction of the shield shell, gravity, thrust from the tail shield, and jacking force. To calculate these forces, it is crucial to obtain accurate geotechnical properties, such as unit weight, cohesion, and effective internal friction angle. However, as previously mentioned, these properties cannot be acquired in real-time during construction and can only be approximated based on pre-construction surveys. Moreover, such investigations can only provide approximate data, which are insufficient to support accurate force calculations. Considering the temporal nature of TBM excavation, the characteristics of the force information, which are difficult to compute directly, can be predicted through temporal sequence analysis. Combining the temporal trends of geotechnical load data during the TBM operation with locally approximated geotechnical properties from surveys enables accurate prediction of the optimal thrust force. The concept of TBM thrust force prediction is shown in Figure 3.3.

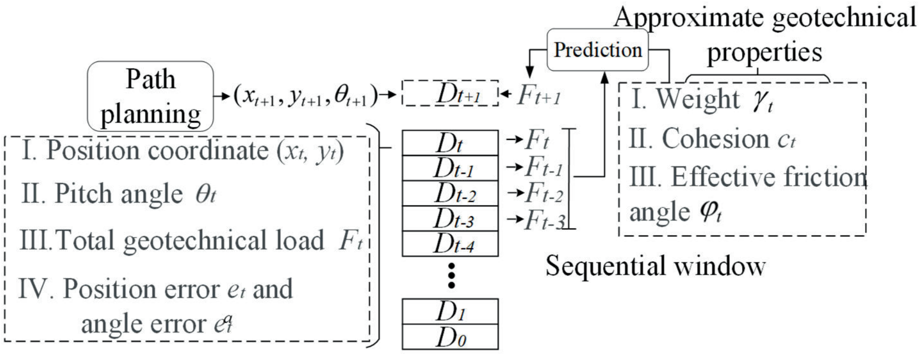


Figure 3.3. The concept of predicting TBM's thrust force.

TBM's operational data is generated in the type of time series. A single entry of data at time step t, denoted as D_t, primarily includes positional data (x_t, y_t), orientation data θ_t, and total geotechnical load F_t. In TBM trajectory navigation, the thrust force must balance the geotechnical load. Therefore, the optimal thrust force F_t^T should be equal to the magnitude of the total geotechnical load F_t. Since the geotechnical properties at the location of the TBM at each moment can only be approximated from pre-construction surveys, which are not accurate, the geotechnical loads at the next timestep F_{t+1} should not be directly computed based on those geotechnical properties. In such a scenario, considering TBM operational data as time-series data $\boldsymbol{D} = \{D_t, D_{t-1}, \cdots, D_0\}$, the changing trend information can be obtained through data analysis with a sliding time window. Moreover, changes in the surrounding geotechnical conditions directly influence variations in TBM thrust. Therefore, geotechnical properties data should also be considered to extract the corresponding trend information. Consequently, the new data F_{t+1} can be predicted using trend information and approximate geotechnical properties. This prediction has the advantage of considering the inherent data changes over time and causal relationships from environmental variations.

3.4.2 PCDM TBM Model

Establishing a TBM motion simulator provides a simulation environment with completed experience space for the RL agent and generates dynamic simulation data for the force prediction model. However, in previous studies, TBM simulators were primarily constructed based on numerical simulation methods such as finite element analysis (FEA) and computational fluid dynamics (CFD). These models have been widely used for predicting TBM–ground interactions and estimating surface settlement. Nevertheless, such models are computationally expensive and therefore unsuitable for sequential decision-making computations commonly required in RL frameworks.

To address this issue, we propose a PCDM of TBM, which discretizes the TBM surface into numerous elemental patches, enabling accurate calculation of local contact forces. These local forces are then aggregated as F_t to estimate the overall thrust F_t^T required for excavation. In the context of TBM trajectory navigation, this model serves as the environment that interacts with the RL agent. During the path planning task at the policy layer, the agent generates actions that are fed into the model, which then outputs the updated TBM position and its deviation from the designed trajectory. Simultaneously, the geotechnical loads acting on TBM along the path are computed, providing the basis for thrust prediction. The structure of the PCDM model is shown in Figure 3.4.

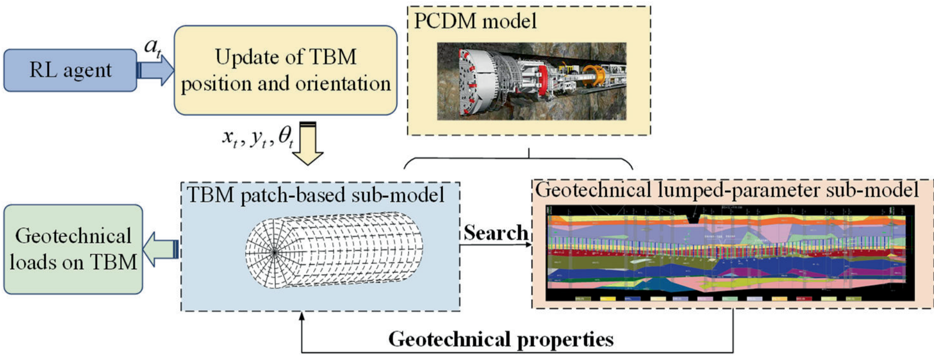


Figure 3.4: Structure of the PCDM model. The model consists of two components: the TBM patch-based sub-model and the geotechnical lumped-parameter sub-model.

In the PCDM model, five main forces affected the advancing movement of the TBM.

(1) Frontal soil resistance F_r, which is the pressure acting on the cutter head from the geotechnical surroundings.

(2) Frictional force between the shield and soil on the shield shell F_f, which is generated

by the mutual compression between the soil and shield, and the relative movement between them. It is influenced by the cohesion of soil layers c_i, the soil weight γ_i, and effective friction angle φ_i.

(3) Jack force, which is a propulsive force from the hydraulic thrust system that acts in the axial direction of the TBM. The jack force is typically divided into four regions: up, down, left, and right, which are F_{ju}, F_{jd}, F_{jl}, and F_{jr}, respectively, and provided independently from the others. The thrust force should be the sum of these forces.

(4) Shield tail friction F_{sf}, which is the force acting between the tunnel lining segment and TBM shield tail generated by the superposition of the pressure from the shield tail wire brush and grease seal.

(5) Rear trailer force F_{RT}, which is the friction between the TBM rear trailing car and the track.

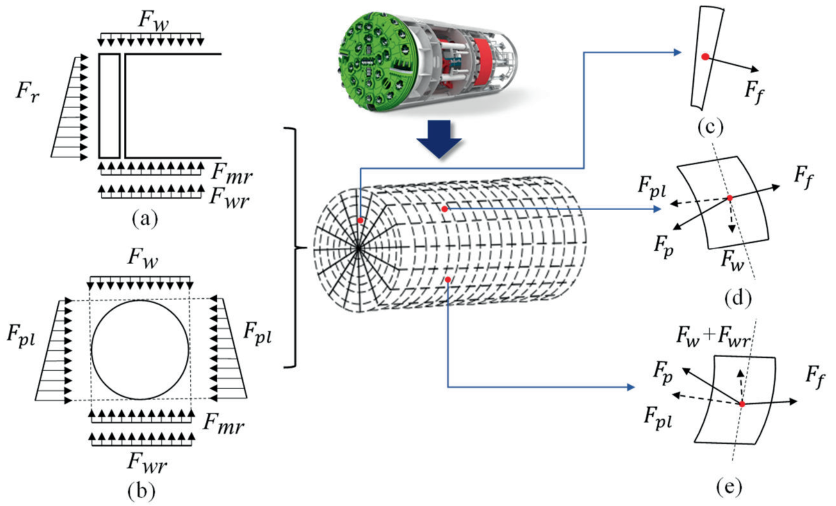


Figure 3.5: The principle of TBM patch-based division and force analysis.

To accurately compute the above forces, a TBM patch-based sub-model is constructed to divide the surface into patches and calculate local forces on each patch. Figure 3.5. illustrates the principle of the TBM patch-based division and the corresponding force analysis in this sub-model. Under this sub-model, the interaction between the machine and soil is considered in three parts with different force conditions: The cutterhead section, which is shown in (a) and (c), experiences forward soil resistance F_r. The upper shield, as shown in (b) and (d), experiences vertical and horizontal soil pressures F_w, F_{pl}, and friction F_f. The lower shield, shown in (b) and (c), experiences soil weight resistance F_{wr}, horizontal pressure F_{pl}, friction F_f, and self-weight resistance F_{mr}. At each discrete patch, the total friction F_{fi} was calculated based on the total frictional force F_{pi}, which was synthesized by the pressure or resistance at the center. The total geotechnical load can be obtained by summing the forces acting on all patches.

In TBM operation, the following condition must be met to ensure forward propulsion:

$$F_t^T = F_{ju} + F_{jd} + F_{jl} + F_{jr} \geq F_r + F_f + F_{sf} + F_{BT} = F_t \tag{3.17}$$

As the total resistance and friction were calculated by summing the forces acting on each patch in the forward direction. The optimal thrust force should be equal to the sum of the geotechnical total load. Eq. (3.17) can be rewritten as follows:

$$F_t^T = F_{ju} + F_{jd} + F_{jl} + F_{jr} = \sum F_{ei} + F_{sf} + F_{BT} = F_t \tag{3.18}$$

where F_{ei} is the force in the forward direction of the i_{th} patch. The thrust force F_t^T that satisfies this condition should be considered optimal.

To facilitate the computation of the forces mentioned above, information concerning the geotechnical conditions in the vicinity of the TBM's placement, which determines soil resistance components, including F_r, F_{wr}, and F_{mr}, as well as frictional force F_f, and pressure forces F_w and F_{pl}, is required. Therefore, a sub-model that can provide the necessary geotechnical information and interact with TBM body is necessary. This information should include details such as the depth, number of soil layers, geotechnical properties of each layer including soil cohesion c_i, soil weight γ_i, effective friction angle φ_i. It is typically obtained through geotechnical surveys and laboratory experiments conducted before the TBM construction phase, and then approximated by interpolation. Based on this information, a geotechnical lumped-parameter sub-model was established in the TBM PCDM model. Based on the geotechnical lumped-parameter sub-model, when the TBM PCDM model is in operation, various geotechnical properties at the center of each patch can be retrieved and used in the computation of geotechnical interaction forces.

3.4.3 Structure of the End-to-end TBM Trajectory Navigation Approach

The structure of the end-to-end TBM trajectory navigation approach is shown in **Figure 3.6**.

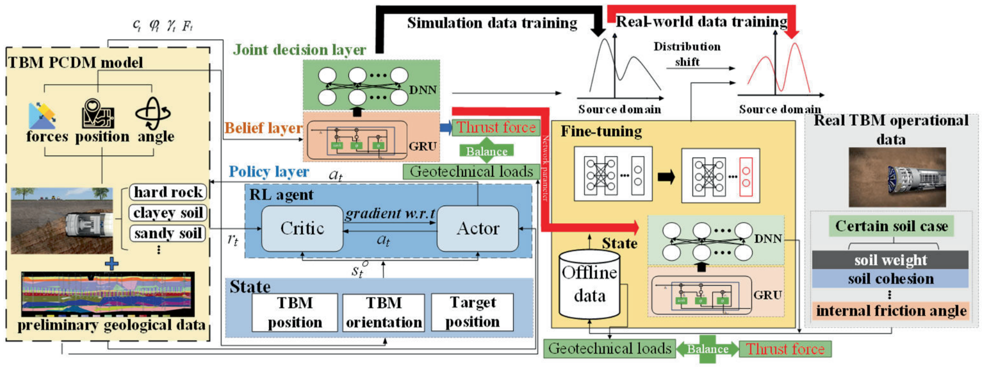


Figure 3.6: Structure of the end-to-end TBM trajectory navigation approach.

In this approach, the RL agent interacts with the TBM PCDM model to facilitate trial-and-error learning and to obtain the optimal path. As the action space of the angle increment is continuous, a stochastic policy is established. In this setup, the actor network outputs the mean and variance of the Gaussian distribution, from which the action is sampled.

Along with the optimal path, the total geotechnical load temporal sequence $\boldsymbol{F}_t$ and geotechnical properties temporal sequence, including c_t, γ_t, and φ_t are collected using PCDM model and fed into a GRU network. The processed hidden states from the GRU are then passed to a DNN to generate the final thrust force prediction $F^T_{t+1}, F^T_{t+2}, \dots, F^T_{t+n}$. The pre-trained CNN and DNN are generalized to practical scenarios by fine-tuning with field data. The structure of the GRU and DNN is illustrated in Figure 3.7.

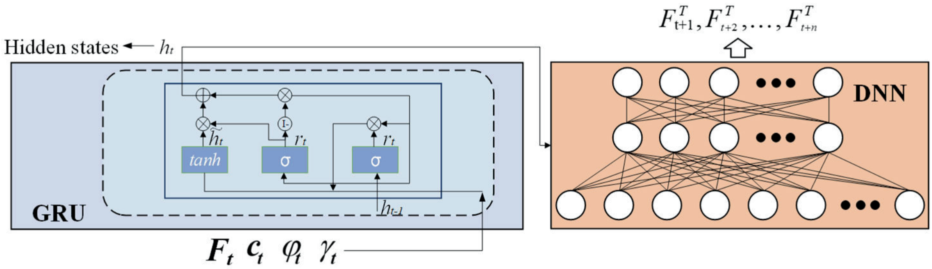


Figure 3.7: The structure of the GRU and DNN.

3.4.4 TBM Trajectory Navigation Approach Design

Based on the designed MDP, the input and output of the approach are defined as follows:

$$\begin{aligned} I_s &= [s_t^o, X_t] \\ O_s &= [a_t, F_{t+1}] \end{aligned} \tag{3.19}$$

The observable state $s_t^o = (\boldsymbol{P}_t, \theta_t, y_t^{df}, \theta_t^{df})$ is provided as input to the RL model. After forward propagation, the action network outputs the expectation and variance of the Gaussian policy distribution. The action is generated with actor policy $\mu(s_t^o \mid \theta^\mu)$, followed by the addition of Gaussian noise $\mathcal{N}$, and finally, it undergoes a sigmoid normalization to obtain the result:

$$a_t = \mu(s_t^o \mid \theta^\mu) + \mathcal{N}_t \tag{3.20}$$

When is applied to the TBM PCDM model, a new state s_{t+1}^o is generated and the critic network evaluates the expected return obtained by action a_t at state s_t^o, which is known as the Q function:

$$Q^\mu(s_t^o, a_t) = E_{(s_t^o, a_t, r_t, s_{t+1}^o) \sim d}[r(s_t^o, a_t) + \gamma Q^\mu(s_{t+1}^o, \mu(s_{t+1}^o))] \tag{3.21}$$

where d denotes the replay buffer. The objective of the critic network is to minimize the following loss function:

$$\mathcal{L} = \frac{1}{N} \sum_i (y_i - Q(S_i^o, a_i \mid \theta^Q))^2 \tag{3.22}$$

The TD target y_i is calculated as follows:

$$y_i = r_i + \gamma Q_i(S_{i+1}^o, a_i \mid \theta^Q) \tag{3.23}$$

where r_i is the immediate reward and Q_t is the Q value obtained from the target critic network.

In this stage, after the optimal path can be obtained by RL training, the $\boldsymbol{F_t}$ c_t, γ_t, and φ_t can also be collected. The multivariate time series M_t, consisting $\boldsymbol{F_t}$ c_t, γ_t, and φ_t, is fed into the GRU model. For each *j*-th gating unit, the update gate z_j^t can be obtained as follows:

$$z_j^t = \sigma(W_z \boldsymbol{M_t} + U_z \boldsymbol{h_{t-1}})^j \tag{3.24}$$

The reset gate g_t^j is:

$$r_t^j = \sigma(W_r \boldsymbol{M_t} + U_r \boldsymbol{h_{t-1}})^j \tag{3.25}$$

Candidate activation $\tilde{h}_t^j$ is calculated as follows:

$$\tilde{h}_t^j = tanh(W\boldsymbol{M_t} + U(\boldsymbol{g_t} \odot \boldsymbol{h_{t-1}}))^j \tag{3.26}$$

Finally, the activation h_t^j is obtained as:

$$h_t^j = (1 - z_t^j)h_{t-1}^j + z_t^j \tilde{h}_t^j \tag{3.27}$$

The hidden state vector $\boldsymbol{h_t}$ was fed into a DNN. Future thrust force F_{t+1}^T can be acquired by training with the loss function

$$Loss = \sum_i (\hat{F}_i^T - F_i^T)^2 \tag{3.28}$$

where $\hat{F}_i^T$ represents the fitted value of the thrust force by the DNN.

3.5 Experiments

In light of the hierarchical POMDP framework and the trajectory navigation approach, two corresponding experiments were conducted: (1) path planning using the deep deterministic policy gradient (DDPG) algorithm in a simulation environment. (2) Force prediction on the TBM using a GRU combined with a DNN based on simulated data and distributional shifting adaptation using offline data. The application effectiveness of the proposed framework was assessed using two critical metrics: (1) the position and angle errors between the TBM trajectory and the designed path in the path planning experiment, and (2) the relative error in thrust prediction experiment.

3.5.1 Path Planning Based on DDPG

First, we evaluated the performance of the proposed approach using the TBM PCDM model. In this experiment, commonly used inclined ascending and descending design trajectories for underground tunnel construction were adopted. The dimensions and operational parameters of the TBM used in the environment are listed in Table 3.1. The empirically tuned optimal hyperparameters for DDPG are summarized in Table 3.2.

Table 3.1: TBM design and operational parameters.

Parameters	Value
Outer diameter	6.9 m
Total length	9.1 m
Total mass	6.7×10^5 kg
Average advancement speed	90 mm/min
Rear support vehicle weight	7×10^5 kg
Segment weight	3.4×10^4 kg

Table 3.2: Hyperparameter setup of DDPG.

Parameters	Value
Input dimension	5
Actor network dimension	$5 \times 128 \times 256 \times 1$
Critic network dimension	$6 \times 128 \times 256 \times 1$
Actor learning rate	1×10^{-4}
Critic learning rate	1×10^{-3}
Batch size	100

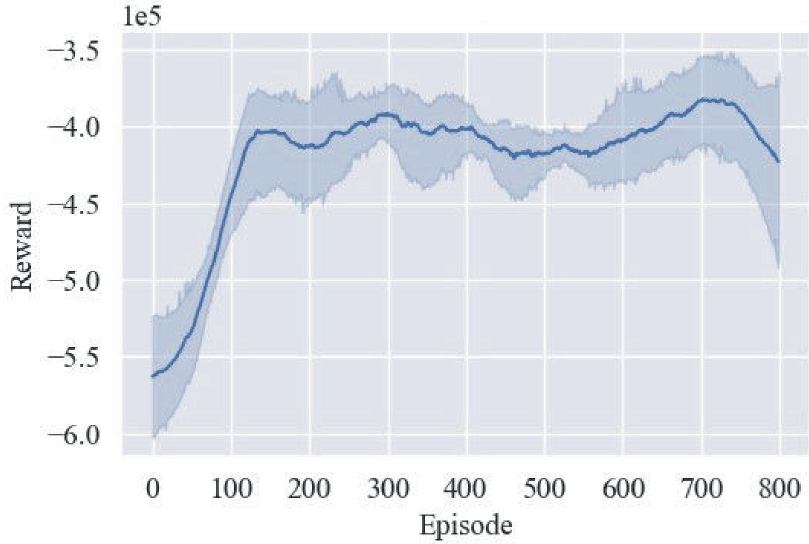
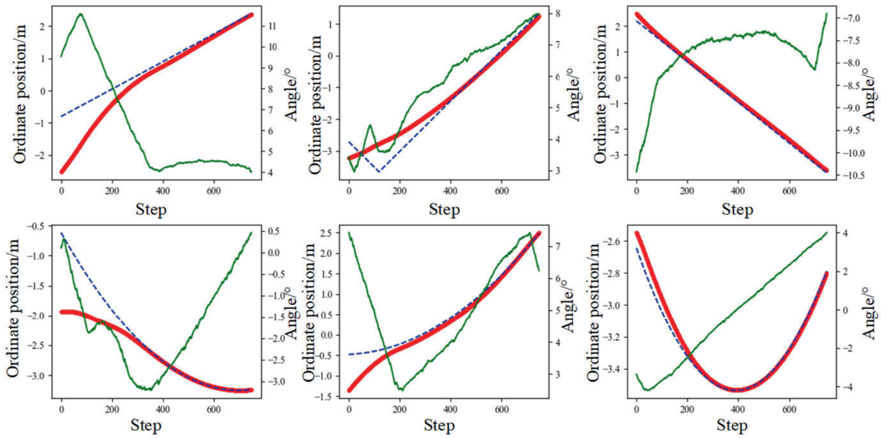
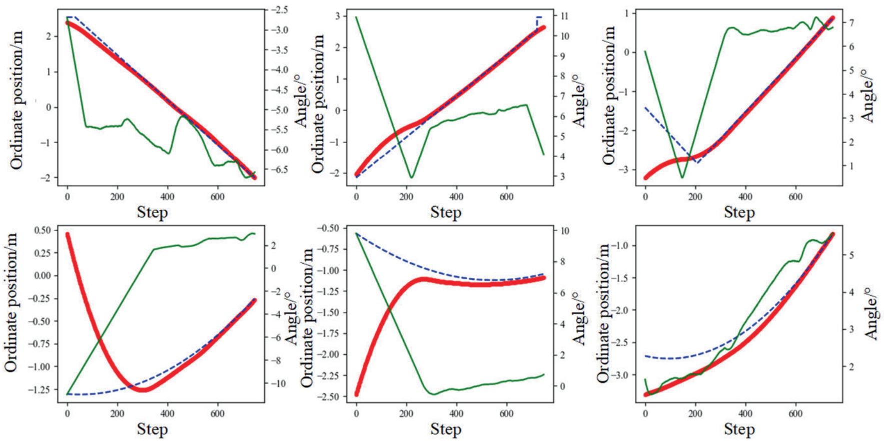


Figure 3.8: Reward curve for TBM path planning based on DDPG.

To match the shape of the designed tunnel in the real world, we randomly generated the design trajectory as a parabola or a V-shaped polyline, with its lowest point uniformly sampled from the coordinate range $y_{lp} = [-1, -3]$. The initial horizontal position coordinate x_{s0}, vertical error y_{e0}, and angle error a_{e0} were uniformly sampled from a specified range: $x_{s0} \sim U(0,65)$, $y_{e0} \sim U(-1.5,1.5)$ and $a_{e0} \sim U(-20^{\circ},20^{\circ})$. The initial vertical coordinate y_{s0} was $y_{s0} = y_{e0} + y_{sd0}$, and the initial angle a_{s0} was defined as $a_{s0} = a_{e0} + a_{sd0}$, where y_{sd0} is the designed vertical coordinate in x_{s0}, and a_{sd0} is the designed angle in the starting point. Under the assumption of a time step duration of 40 s, an episode consists of 750 steps, and the total straight-line distance traveled by the TBM is 45 m. The trend of the reward function is shown in Figure 3.8, and the selected representative RL agent path planning performances and search capability of the RL agents are depicted in Figure 3.9 and 3.10, respectively.



(a)



(b)

Figure 3.9: TBM path planning using DDPG from different initial points. (a) Performance of the RL agent after training for four million timesteps in path planning. (b) Performance when the trained agent is used in another environment $y_{lp} = [0,-1] \cup [-3,-4]$. In (b), the agent can succeed in an unseen environment because both the TBM and target positions and angles are included in the state. This allows the RL agent to learn to reduce the error between these two factors during the training process, rather than by simply following a specific curve. This enhances its generalization ability. (In each subplot, the red curve represents the TBM's motion trajectory, the blue dashed line represents the design trajectory, and the green curve represents the pitch angle.)

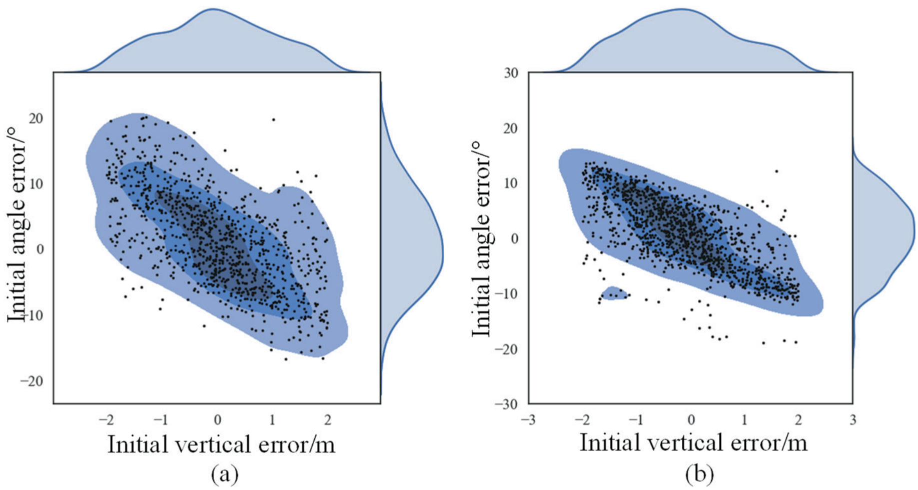


Figure 3.10: Joint distributions of initial point errors for successful trajectories in path planning using RL. (a) Distribution of V-shaped polylines, (b) Parabolas. In the graph, black dots represent successfully planned trajectories. The horizontal and vertical coordinates denote the initial trajectory's vertical and angular errors concerning the target trajectory. The curves above and to the right of each scatter plot depict the distribution of dots along the horizontal and vertical axes.

From Figure 3.8 and 3.9, it is evident that, through the training process, the RL agent eventually converges to a distribution that yields the maximum reward under which the optimal path can be found in most initial cases. These successful cases achieved average position and angle errors of 0.007 m and 0.415°, respectively, surpassing the majority of manually controlled results based on empirical expertise. However, it should be emphasized that not all initial states necessarily lead to successful path planning. The correction capability of the RL agent is limited by the error between the starting point and design trajectory in each episode. This limitation is reflected in the density distribution of the successful path planning initial points, as shown in Figure 3.10. The success rate of the path planning increases as the initial angle a_{e0} and vertical error y_{e0} decrease, which is intuitive. In addition to this trend, higher success rates were observed under two specific scenarios: $a_{e0} > 0$ and $y_{e0} < 0$, as well as $a_{e0} < 0$ and $y_{e0} > 0$. These conditions bring the correction trajectory closer to the optimal realignment angle than in the other areas. A similar principle can be applied to the scenario where the initial angle error is below zero and the initial vertical error is more significant than zero. This range of correction capabilities mirrors the situations encountered in the construction processes.

To validate the effectiveness of RL in TBM path planning, we select the heuristic greedy algorithm as the baseline method to conduct a comparative experiment employing. All simulation environments and TBM parameter configurations remained consistent with those used in the RL path planning. Joint distributions of initial point errors for successful trajectories in path planning using the greedy algorithm are shown in Figure 3.11.

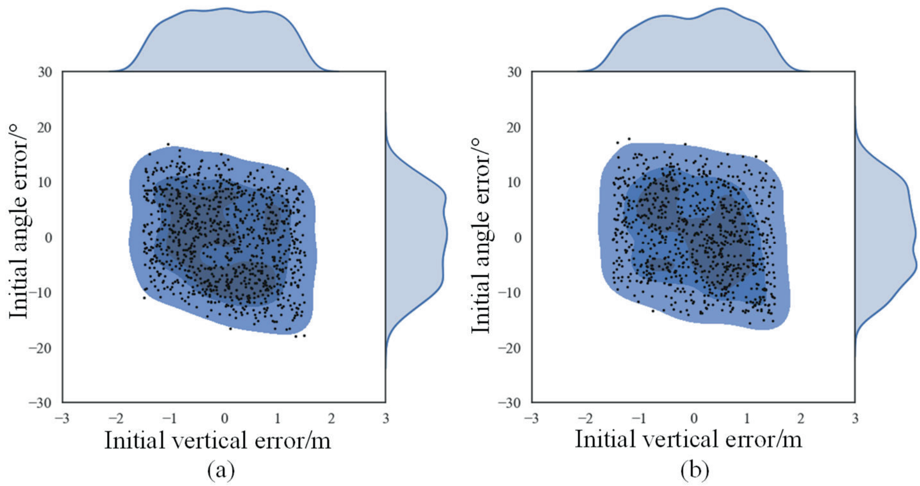


Figure 3.11: Joint distributions of initial point errors for successful trajectories in path planning using greedy algorithm. (a) Distribution of V-shaped polylines, (b) Parabolas.

The greedy algorithm remains capable of achieving successful path planning from certain random initial points. However, Figure 3.11 clearly demonstrates that the distribution of initial errors in successful trajectory cases is more concentrated. Compared to the diamond-shaped distribution observed in RL, the greedy algorithm fails to handle initial states characterized by either large vertical errors or significant angular errors. These specific initial conditions are more prone to cause trajectory deviations from the designed path. This result indicates a higher probability of failure when employing the greedy algorithm under such circumstances. The success of RL in handling these challenging initial states substantiates its superior path planning capability in this scenario.

Beyond the V-shaped and parabolic trajectories, extensive validation has been conducted on semi-elliptical, polyline, W-shaped, and double-sigmoid trajectories, all of which consistently demonstrate successful path planning performance. These results substantiate the strong scalability and wide applicability of the proposed RL agent in TBM path planning applications. Nevertheless, owing to the limited utilization of such trajectory patterns in practical engineering scenarios, a detailed discussion of these experimental results is not included in this study.

3.5.2 Thrust Force Prediction and Transfer Learning with Real-world Data

Once TBM path planning is achieved, this foundation can be used to implement real-time predictions during TBM operation. The GRU extracts features from the total geotechnical load sequence of the TBM advancement process. These extracted features and the geotechnical information of the current location were then fed into a DNN to generate the final predicted thrust force values in the forward direction of the TBM.

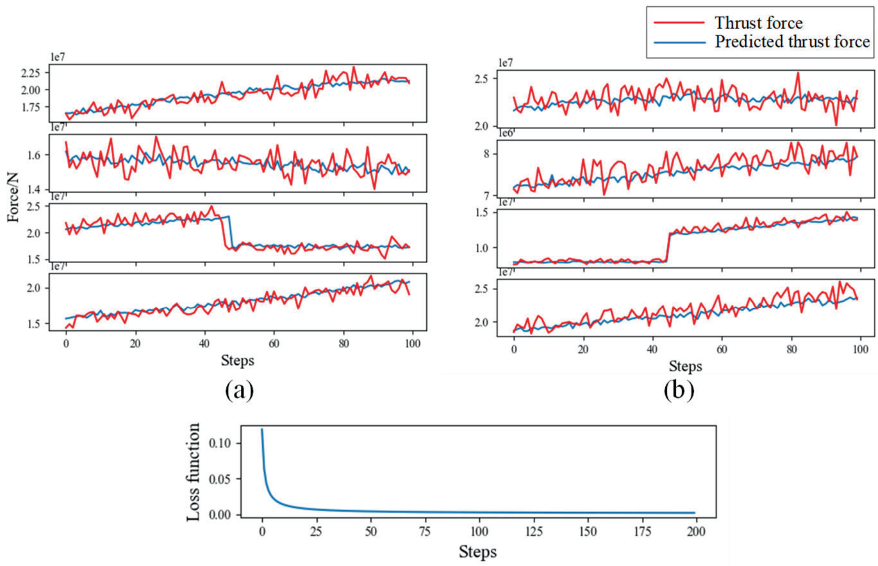
This study validates the effectiveness of the proposed method using data simulated and collected from two distinct geotechnical conditions. The first environment predominantly features weathered rock strata, including gneiss, schist, completely weathered granite, and highly weathered granite. Characterized by high yet variable strength, substantial bearing capacity, and heterogeneous hardness, this configuration simulates the subsurface geotechnical conditions typical of coastal cities in southern China. We designate this geotechnical environment as *Geotechnical Environment A*, with the corresponding thrust dataset collected in the construction under these conditions referred to as *Thrust Dataset A*.

The second environment consists primarily of unconsolidated sedimentary layers, mainly composed of clay, silty clay, silt, and sand strata. This geological profile exhibits high water content, significant compressibility, low strength, elevated sensitivity, and notable rheological behavior, representing subsurface conditions commonly encountered in coastal cities of northern China. We designate this geotechnical environment as *Geotechnical Environment B*, with the corresponding thrust dataset collected in the construction under these conditions referred to as *Thrust Dataset B*.

The empirically tuned optimal hyperparameters for both the GRU and DNN architectures used in our experiments are summarized in Table 3.3. The hyperparameter settings of the model are entirely derived from experimental experience. The selected training results are presented in Figure 3.12.

Table 3.3: Hyperparameter setup of GRU and DNN.

Parameters	Value
Input dimension	4
GRU layers	2
Hidden size	32
DNN dimension	32 × 128× 64× 1
Learning rate	1×10^{-3}
Batch size	100



(c)

Figure 3.12: Loss variation and thrust force prediction curves using GRU. (a) Comparative results

between predicted thrust curves and simulated measurements in *Geotechnical Environment A*. (b) Comparative results between predicted thrust curves and simulated measurements in *Geotechnical Environment B*. (c) GRU loss function convergence curve.

Figure 3.12 shows that the GRU model achieved a good fit and the predicted results after training were accurate. The total error between the predicted and actual thrust force curves was 3.93%, which is acceptable in an engineering context. This can be attributed not only to the high accuracy of the GRU in predicting time-series data, such as geotechnical information, but also to the direct impact of geotechnical properties as feature inputs. The thrust force prediction performance during the path planning process is illustrated in Figure 3.13.

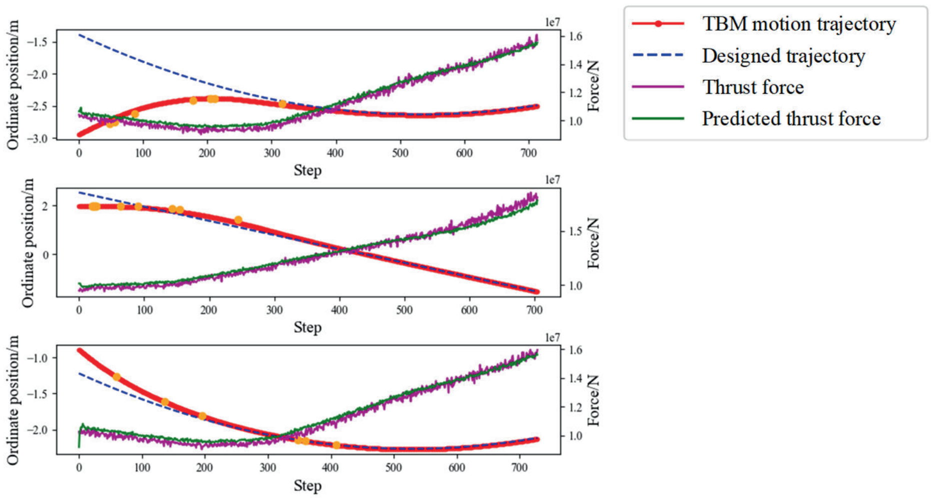


Figure 3.13: Performance of force prediction during the path- planning process.

From the graph, we can see that the TBM forces can also be predicted accurately in an accurate path-planning process. During this process, only a few points had a relative error of more than 10% compared with the true values, as indicated by the orange points in the graph. This occurred for less than 1% of the entire running process (approximately 40-m long). The above experimental results demonstrate that by employing GRU to construct the belief layer for belief state inference and feeding the distribution of unobservable states into the DNN, accurate thrust prediction can be achieved during the majority of the TBM's operational period.

To investigate the performance of different temporal prediction network architectures in constructing the belief layer, we selected Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) as benchmark comparisons to evaluate against the prediction performance of the GRU model. In our experiments with RNN and LSTM, we maintained network architecture parameters consistent with those of the GRU network. The specific hyperparameters setup are illustrated in Table 3.4. The prediction results after training the RNN and LSTM networks on both datasets simulated in *Geotechnical Environment A* and *B* are presented in Figure 3.14 and Figure 3.15, respectively.

Table 3.4: Hyperparameter setup of RNN and LSTM.

Parameters	RNN	LSTM
Input dimension	4	4
Number of layers	2	2
Hidden size	32	32
Learning rate	1×10^{-3}	1×10^{-3}
Batch size	100	100

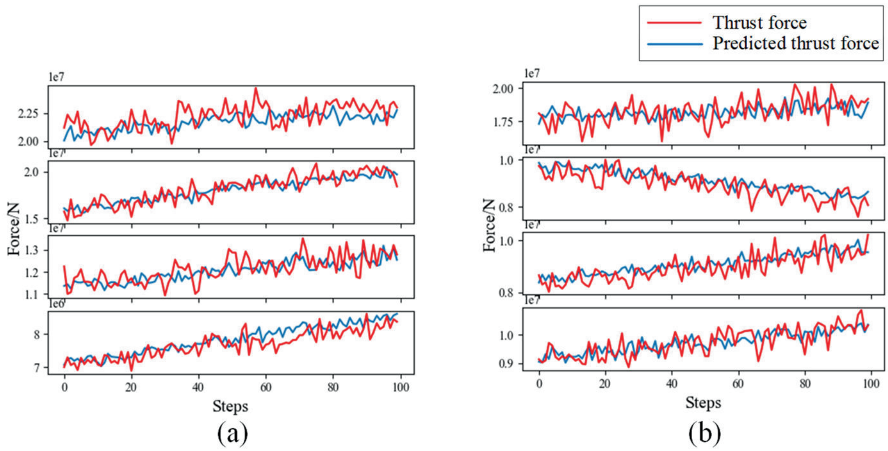


Figure 3.14: Comparative results between predicted thrust force curves using RNN and actual measurements. (a) Dataset simulated in *Geotechnical Environment A*, (b) Data simulated in *Geotechnical Environment B*.

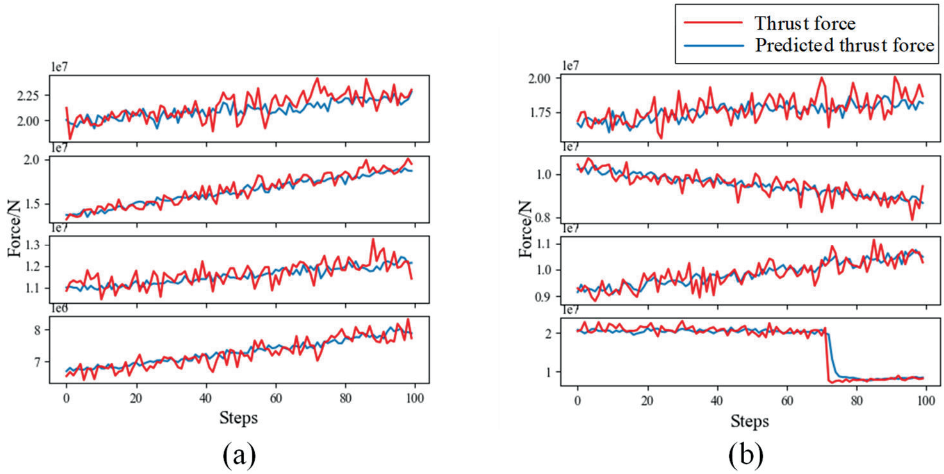


Figure 3.15: Comparative results between predicted thrust force curves using LSTM and actual measurements. (a) Dataset simulated in *Geotechnical Environment A*, (b) Data simulated in *Geotechnical Environment B*.

The RNN architecture achieved a prediction error of 4.24%, while the LSTM model yielded an error of 6.33%. In comparison with these two networks, the GRU attained a lower error rate of 3.93%, demonstrating its superior performance in this application.

A box plot analysis was conducted to examine the error distributions in thrust prediction following belief state construction across these three network architectures. The results are presented in Figure 3.16.

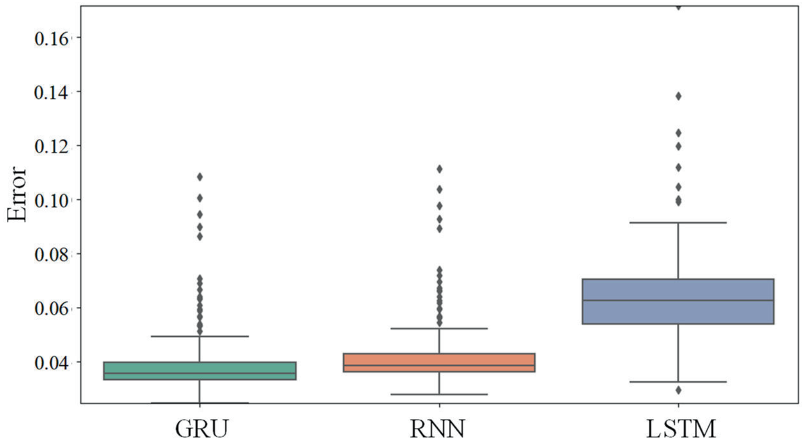


Figure 3.16: Box plots of thrust prediction error distributions obtained using GRU, RNN, and LSTM for belief layer construction.

As evidenced by the Figure 3.16, the GRU network demonstrates superior performance compared to the other two architectures. These results confirm GRU's exceptional capability in TBM thrust prediction.

Transfer learning experiments with real-world data

The distribution of the trained prediction model was transferred via a fine-tuning method using real-world data to validate its generalization capabilities. The real-world data used in the study were *Thrust Dataset A* and *B*. Among these, *Thrust Dataset A* was obtained from a subway construction project in southern China, with a construction length of 1641.241 m. *Thrust Dataset B* was collected from a metro construction project in a coastal city of northern China, with a construction length of 669.736 m.

The fine-tuning experiment results are presented in Figure 3.17. The overall mean error across both datasets stands at 5.84%. Although these errors show some increase compared to those observed in simulation datasets, they remain within an acceptable range for practical applications. It can be concluded that this model can provide accurate predictions when applied to similar real-world environments. Although different uncertainties and disturbances exist in various environments, the model can successfully predict the trends of the changes.

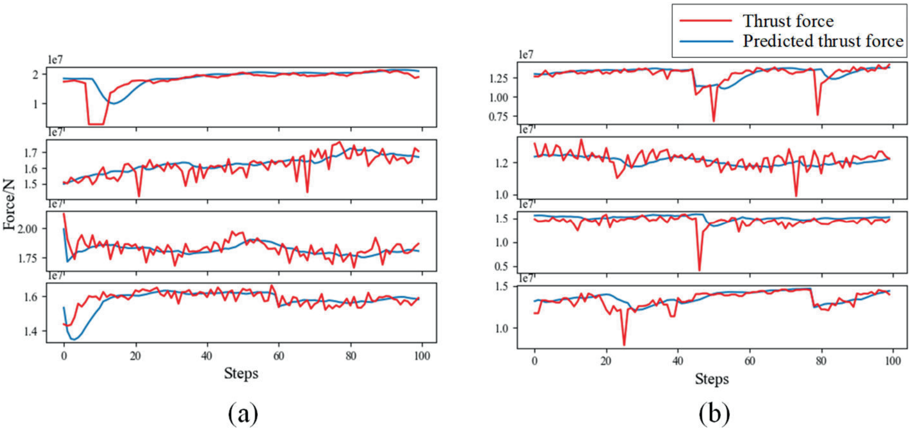


Figure 3.17: Performance of thrust force prediction after fine-tuning using real-world data using *Thrust Dataset A* and *B*. (a) *Thrust Dataset A* is utilized. (b) *Thrust Dataset B* is utilized.

The experimental results demonstrate that applying the proposed hierarchical POMDP framework enables accurate decision-making in TBM trajectory navigation. Comparative studies confirm that the selected architectural components - RL for the policy layer, GRU for the belief layer, and DNN for the joint decision layer - form a rational and effective implementation. Furthermore, the framework exhibits strong generalization capability across varying operational conditions.

3.6 Summary

This study aims to solve the TBM trajectory navigation problem despite incomplete geotechnical information. Since real-time geotechnical properties for thrust force prediction cannot be directly measured, we proposed a hierarchical POMDP framework. Within this framework, a PCDM model is firstly developed to simulate motion and force mechanics of TBM. Subsequently, an RL agent is employed in the policy layer to plan the optimal excavation path by leveraging observable information through interaction with the PCDM model. Then, in the belief layer, a GRU network is used to infer the belief distribution of unobservable geotechnical properties by analyzing the TBM's state time series along the optimal path. Finally, the joint decision layer outputs the optimal thrust. Experimental results demonstrate that this approach can accurately predict the required thrust in both simulated and real-world data, enabling the TBM to excavate along the optimal trajectory.

In this study, an RL agent was trained in a simulated environment with various design trajectory shapes and tested on different shapes to evaluate generalization. Results showed the RL agent could find the optimal path within a range of initial positions and angular errors and successfully execute path planning tasks in new environments. The thrust force prediction, evaluated using simulated geotechnical data, demonstrated high accuracy. Transfer learning confirmed the approach's robust generalization to real-world data. Our experiments demonstrated that both the RL agent and GRU model are reliable and generalizable. Integrating RL and time-series prediction provided an end-to-end decision-making approach, suitable for real-world excavation testing.

The hierarchical POMDP framework demonstrates significant advantages through three key aspects. First, by fully leveraging causal relationships between observable and unobservable states, it can mitigate decision errors caused by incomplete observations. Furthermore, it transforms the traditional serial belief inference into a parallel architecture that shortens error propagation paths and minimizes error during decision-making. Finally, by decomposing the original POMDP into three specialized layers, it effectively reduces problem complexity while simultaneously enhancing the method's flexibility, scalability, and generalization capabilities. However, the potential limitation lies in the fact that the framework's performance strongly depends on the temporal coherence of state transitions and the accuracy of belief-state estimation. When temporal dependencies are weak or the inferred belief states are biased, error propagation may occur, ultimately compromising the reliability and robustness of downstream decision-making.