Article citation info:
Liu J, Min W, Zhong Y, Luo A. Research and application of obstacle avoidance algorithm for hydropower station inspection robot based on spatio-temporal networks. Diagnostyka. 2026;27(1):2026102. http://dx.doi.org/10.29354/diag/217181.

DIAGNOSTYKA, 2026, Vol. 27, No. 1

e-ISSN 2449-5220
DOI: 10.29354/diag/217181

RESEARCH AND APPLICATION OF OBSTACLE AVOIDANCE ALGORITHM FOR HYDROPOWER STATION INSPECTION ROBOT BASED ON SPATIO-TEMPORAL NETWORKS

Jiajia LIU, Wanxiong MIN *, Yuanfeng ZHONG, An LUO

Guizhou Wujiang Hydropower Development Co., Ltd., Goupitan Power Plant, Zunyi, 564400, China
*Corresponding author's e-mail: wanxiong_min@outlook.com

Abstract

The inspection of hydropower stations using autonomous robots is vital for ensuring operational safety and efficiency. Obstacle avoidance plays a crucial role in enabling these robots to navigate complex environments that are filled with both static and dynamic hazards. However, existing obstacle avoidance methods often struggle to handle real-time changes in spatial and temporal contexts, resulting in suboptimal path planning and an increased risk of collision. To address these limitations, this research proposes a Spatio-Temporal Graph Convolutional Network (ST-GCN) based framework that effectively models both the spatial layout and dynamic movements of obstacles over time. The proposed ST-GCN framework processes real-time sensor data (LiDAR, cameras, IMU) and historical movement patterns to predict obstacle trajectories and adapt the robot's navigation path accordingly. This approach allows the inspection robot to dynamically adjust its route in environments such as turbine halls, where moving machinery and personnel are common. Experimental evaluations in a simulated hydropower station environment demonstrated that the ST-GCN-based method significantly outperformed traditional reactive models, achieving higher accuracy in obstacle prediction and safer, more efficient navigation. These findings validate the effectiveness of spatio-temporal modeling for intelligent obstacle avoidance in industrial robotic inspection tasks.

Keywords: obstacle avoidance, ST-GCN, hydropower station, inspection robot, path planning

1. PROLOGUE

In modern hydropower stations, it is crucial to deploy autonomous inspection robots to ensure that operations are safe and efficient [1]. In these environments, which are both dynamic and full of moving machinery, people, and static obstacles, it's challenging to avoid obstacles because most conventional systems rely on reactive mechanisms that don't account for changes in space and time in real-time [2]. ST-GCN does a great job at capturing both the layout of the environment and how dynamic obstacles change over time [3]. The suggested methodology lets the robot change its navigation course ahead of time to avoid known dangers using real-time data from sensors like LiDAR, cameras, and inertial measurement units (IMU) along with learnt patterns from past movement paths [4]. This makes it possible to accurately predict how obstacles will move [5]. The robot's safety and efficiency are considerably improved when it can predict how obstacles will behave [6]. This is especially important in high-risk areas like turbine halls, where unexpected changes in behavior can lead to accidents or delays [7]. The system is much better at handling surprises because it can take in both real-time environmental data and patterns acquired from past activities[8]. The study suggests that industrial robotics can benefit from deep learning methods utilizing an ST-GCN-based technique that extends beyond merely avoiding obstacles, offering a solution for the future [9]. According to tests conducted at a simulated hydropower facility, the proposed approach enhances both trajectory predictions and real-time navigation [10]. These results support the hypothesis that intelligent robots could benefit from employing spatio-temporal networks in general, and especially for checking infrastructure that is important for public safety.

1.1 Problem statement

It is well known that typical obstacle avoidance algorithms for inspection robots struggle to operate effectively in hydropower stations. There are both permanent structures and moving obstacles at these stations, such as people and machines. These rules make it harder to find the way around and make it more likely that will crash into anything. There is a need for better prediction system that can deal with changes in space and time.

In the last few years, a number of predictive and learning-based obstacle avoidance models have been

Received 2025-07-23; Accepted 2026-01-20; Available online 2026-01-21

developed. However, most of them can't simulate spatial relationships and temporal evolution together in industrial settings that change quickly. Most current methods deal with spatial and temporal data separately, which means they don't respond quickly to changes in barrier behavior. This is especially true in hydropower stations, where people, spinning gear, and mobile equipment create constant, non-linear motion patterns. Current prediction-driven approaches also depend on fixed temporal frames or static representations, which limits their ability to change the paths of obstacles when there are unexpected, short-term changes.

The gap this work addresses is the lack of a single spatio-temporal learning framework that can capture both the structural arrangement of hydropower station surroundings and the changing movement patterns of various barriers that interact with each other. The suggested ST-GCN model closes this gap by putting geographic closeness, inter-object impact, and time-dependent motion transitions into one graph-based representation. This lets the system predict the paths of obstacles more accurately and change navigation decisions in real time. The technology offers a more flexible, context-sensitive prediction system than traditional learning models. This makes it especially useful for hydropower inspection situations when environmental conditions change without warning.

1.2 Motivation

Self-driving inspection Robots that work in hydropower plants have a lot of trouble getting around because machines and people move in ways that are hard to predict. Our research attempts to make these environments safer and more efficient for robots. In such situations, smart, real-time obstacle avoidance and adaptive course planning are crucial, but the solutions are often inadequate. A spatio-temporal, predictive strategy is needed.

1.3 Contributions

This study presents a novel approach for inspection robots at hydropower stations to navigate around obstacles using Spatio-Temporal Graph Convolutional Networks (ST-GCN). By providing accurate predictions of obstacle trajectories and dynamically adjusting navigation paths in real-time based on sensor inputs and historical movement data, the technology enhances the safety, efficiency, and adaptability of complex industrial settings.

The remaining section of this paper is set up like this: In Section 2, this paper look at relevant studies and strategies that have been suggested before. In Section 3, paper discussed about the proposed framework for avoiding obstacles that uses Spatio-Temporal Graph Convolutional Networks (ST-GCN). Section 4 compares the proposed method to traditional ones. Section 5 wraps up the paper and talks about possible areas for future research.

2. RELATED WORKS

This section is about intelligent obstacle avoidance. It discusses the latest research in spatio-temporal modeling, machine learning, and robotic vision. This study proposes an inspection robot framework based on ST-GCN, which considers the complex and dynamic nature of hydropower plant environments. It provides guidance on deep learning methods, sensor integration, environmental issue prediction, and control strategies.

This study by Wang et al. [11] presents a deep learning framework for tactile object classification (DLF-TOC) to make robotic hands smarter and more agile for use in smart industries. It utilizes convolutional neural networks to comprehend sensory information and facilitate accurate object recognition. Although it does not focus on navigation, it's useful for automated inspection jobs, as it helps robots interact with their surroundings and be aware of their environment.

This study examines the temperature changes on the surface of Arctic periglacial habitats, utilizing both thermal images from unmanned aerial vehicles and ground-based measurements by Alphonse et al. [12]. This result highlights the importance of data fusion and spatio-temporal sensing in monitoring the environment. This study doesn't directly focus on robotics, but it does lay the groundwork for combining data from multiple sources. This will be particularly helpful for inspection robots that must operate in unpredictable and challenging outdoor environments.

Author Tu [13] examines the intelligent hydraulic structure of the Dadu River hydropower system, utilizing data-driven management and AI-powered predictive maintenance and optimization. This study demonstrates the importance of incorporating innovative systems and real-time control into water infrastructure. It does not directly address mobile robots, but its methods do aid in the development of autonomous monitoring and flaw detection systems related to hydropower.

This conference paper by Mekuria et al. [14] discusses the use of robots that operate underwater and in the air to monitor freshwater lakes in Africa. It demonstrates how autonomous systems can be utilized in the real world to protect the environment by integrating low-cost deployment with spatio-temporal data collection. Its main focus is on environmental issues, and there are many similarities to the work of inspection robots, especially when it comes to using sensors and navigating independently in changing natural environments.

Author Hożyń [15] provides detailed information on how to recognize visual movements in underwater interactions between people and robots. This research analyzed how far vision, machine learning, and recognition algorithms have progressed in the last few years, especially for applications in dark and noisy environments. Gestures are the primary focus, and perception

models and environmental resilience enable underwater inspection robots to navigate and avoid obstacles.

This research by Tholen et al. [16] presents a low-cost multi-sensor platform (LcMsP) for studying the spatio-temporal outflow of groundwater that is buried. This is an example of how integrated sensing technology can be utilized in the ocean to monitor changing conditions. The study doesn't directly discuss robotics, but it does provide some ideas on how to design and utilize sensors in challenging and unexpected environments, such as hydropower facilities, where aquatic inspection robots can utilize them.

Researchers Agonafir and Zheng [17] examine the complexity of spatio-temporal machine learning models (StMLMs) in predicting urban floods. It offers structured evaluation methods to enhance the accuracy of forecasts while maintaining low computing costs. Its methods for dynamic forecasting and real-time environmental modeling enable inspection robots to navigate areas prone to floods or hydraulically active environments more safely, even if they aren't directly related to robotics.

Li et al. [18] present a method for estimating river levels using spatio-temporal correlations reliably. Their combined model produces more accurate forecasts, even in cases of ambiguity in the water cycle. Their main focus is on monitoring the environment, and their method may be utilized for self-driving cars in areas where topography changes, such as hydropower facilities where real-time changes in water levels affect route safety and decision-making.

This research He et al. [19], looks at different ways to operate autonomous underwater vehicles, focusing on formation control, trajectory tracking, and path following. It compiles the most up-to-date methods for navigating complex aquatic environments. This study has direct real-world applications for hydropower inspection robots, particularly those that must navigate complex, dynamic environments with numerous obstacles, utilizing adaptive path planning and reliable motion control.

The paper by Islam et al. [20] used computer vision to study oceanography and underwater robotics. They discuss how to find things, create maps, and avoid trouble. They discuss issues with visibility underwater and recommend employing multimodal sensing and image-based algorithms to address them. This information helps create robots that can inspect underwater hydropower plants and feature dependable navigation systems that enhance their vision.

The suggested ST-GCN-based obstacle avoidance system proved to be better than previous systems in that it accurately models the physical configurations of obstacles, and additionally provides for the evolving characteristics of each obstacle, as it related to the passage of time. Unlike static models or reactive models, this model is adaptive and supports immediate real-time predictions of the path. The result permits inspections of complex hydropower infrastructures to be safer and more thorough and efficient, because it overcomes the issues of visibility and route planning.

To offer a more integrated viewpoint, the examination of current methodologies can be organized into three major categories: sensor-driven perception models, spatio-temporal prediction frameworks, and robotic navigation and control systems. Sensor-driven methods mainly aim to improve how we perceive our surroundings using tactile sensing, thermal imaging, or multi-sensor fusion. However, they don't look at how the behavior of moving obstacles changes over time, which makes them less effective for real-time navigation in hydropower plants. Spatio-temporal prediction models enhance forecasting by learning movement dependencies; however, many of these methods address spatial layouts and temporal transitions independently or depend on superficial temporal modeling, leading to diminished forecasting accuracy during abrupt environmental shifts. Studies focused on navigation examine trajectory tracking and path-following techniques; nevertheless, they frequently presuppose stable surroundings or predictable object motion, rendering them inadequate for contexts characterized by continual human-machine interactions.

The gap discussed here is due to the absence of a unified framework that can learn spatial relationships, temporal dynamics, and obstacle interaction patterns all at once in complicated industrial settings. The suggested ST-GCN method fixes this by making a single spatio-temporal graph representation that shows closeness, motion continuity, and inter-object influence all at the same time. This synthesis helps the model get around the problems with earlier methods, which lets it forecast the paths of obstacles more accurately and alter its plans when inspecting hydropower plants. Related works summary is shown in Table 1.

3. PROPOSED SECTION

This method enables powerful tools for hydropower inspection robots to counter barriers in real-time with spatio-temporal vision combined with predictive modeling. The system uses sensor fusion, feature encoding, and ST-GCNs to predict the expected motions for obstacles and change course in real-time. The design integrates optimization, graph-based learning, and theoretical modeling to qualify the navigation process safely and effectively through complex industrial spaces.

Table 1. Related works summary

S.No	Methods	Advantages	Limitations
1	Deep learning for tactile object classification in robotic hands[11]	Enhances robotic manipulation and object recognition accuracy in smart factories	Focuses on classification, not navigation or spatio-temporal dynamics
2	UAV thermal imaging + in-situ data analysis for temperature variability[12]	Provides high-resolution, real-time environmental data with spatial and temporal insights	Not robot-centric; limited to static thermal monitoring applications
3	Intelligent control and monitoring of hydraulic structures[13]	Supports predictive maintenance and AI-based decision-making in hydropower systems	Does not involve mobile robots or navigation techniques
4	Drone and underwater robot integration for lake monitoring [14]	Demonstrates real-world use of autonomous systems for environmental surveillance	Emphasizes conservation, not complex obstacle avoidance or real-time path planning
5	Visual gesture recognition using machine learning[15]	Improves underwater HRI, with robust visual perception in challenging aquatic conditions	Focused on gesture commands, not navigation or multi-obstacle environments
6	Multi-sensor platform for spatio-temporal marine data collection[16]	Enables low-cost environmental monitoring in marine settings	Not optimized for mobile or autonomous robotic platforms
7	ML-based urban flood forecasting with spatio-temporal modelling[17]	Enhances prediction accuracy with balanced model complexity	Applied to environmental forecasting, not robotic navigation
8	Integrated spatio-temporal river level prediction model [18]	Accurate water-level forecasting is beneficial to risk-sensitive domains	Lacks real-time robot interaction or adaptation capabilities
9	Review of AUV path following and control strategies[19]	Comprehensive summary of trajectory tracking and formation control in underwater robotics	Mainly theoretical; implementation in varied hydropower terrains not demonstrated
10	Computer vision for underwater robotics[20]	Covers perception, mapping, and obstacle detection in submerged environments	Primarily review-based; does not propose or test a specific navigation framework.

3.1 Proposed System Architecture Overview

A system architecture that uses multiple types of sensors integrated with machine learning algorithms allows autonomous inspection robots to avoid obstacles with a high level of precision. The Sensor Module is the initial step in the process. It collects data from the LiDAR, IMU and cameras in real-time. Therefore, the robot can detect its environment [21].

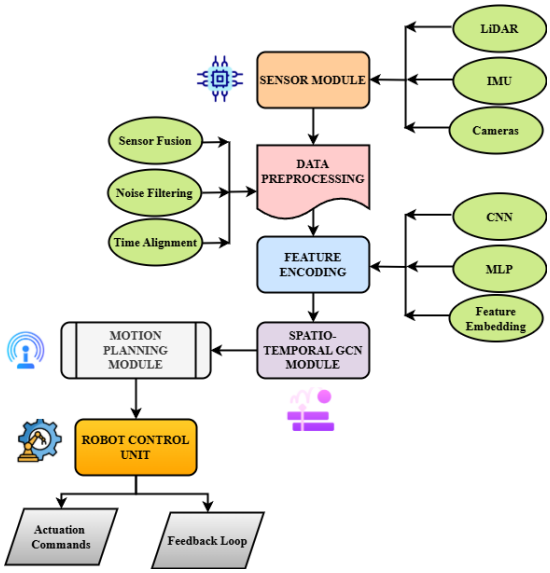


Fig. 1. Proposed system overview

To ensure the input is correct, the raw data is preprocessed using methods that include synchronizing timestamps, filtering, and removing noise, as shown in Figure 1. In the next phase, Feature Encoding employ CNNs and MLPs to represent the cleaned data in a more general way. Once these properties have been put into graphs, the (ST-GCN) represents both the spatial interactions (such as distances and proximities) and the temporal dynamics (like how obstacles move). For moving environmental entities, the output is a set of expected paths. The Motion Planning Module utilizes these predictions to design paths that are both safe and efficient in real-time.

The environment is divided into a spatio-temporal graph, where each detected entity is a node and the edges show how the entities are related in space or behavior. Nodes represent both static structures (such walls, turbine casings, and pillars) and dynamic impediments (like workers, rotating machinery, and moving equipment). Each node is encoded utilizing fused LiDAR, IMU, and camera characteristics. A distance-based adjacency rule is used to make spatial edges. This rule says that objects within a certain interaction radius share connection. This lets the model show how close items are to each other and how they could collide. Temporal edges connect the same item at different points in time, which lets you learn about changes in speed, patterns of acceleration, and short-term changes in motion.

This framework uses an ST-GCN architecture made up of stacked spatial-temporal convolutional layers that function on both adjacency matrices and

sequences of temporal features. Spatial convolution gathers information from nearby nodes to record local interactions, whereas temporal convolution pulls out motion progression that is stored along the time axis. There are graph convolution, batch normalization, and ReLU activation in each block. Then, there are residual connections to make deep spatio-temporal learning more stable. The last layer's output gives future obstacle trajectory states for a set prediction horizon.

The training approach uses supervised learning, which means that ground-truth trajectory labels and synchronized sensor records are turned into graph sequences. A combined reconstruction and trajectory-prediction loss punishes differences between predicted and actual motion. This lets the network improve its internal spatio-temporal representations. We use Adam with a planned learning rate decay for optimization, and we train on GPU-enabled environments to make it easier to interpret high-dimensional sensor streams.

The ST-GCN works with streaming sensor data at 25 to 30 frames per second for real-time use. Because the graph convolution blocks are light and the feature dimensionality is lower after fusion, each inference step only takes a few milliseconds. This makes it appropriate for onboard processing modules used in inspection robots, which means that predicting the trajectory of an obstruction and designing a new path may be done without any delays in hydropower station surroundings.

The Control Unit then follows the motion commands and sends new information about the surroundings to the system where it can learn and change over time.

$$Ft(ip) = s_s + S_p L_{yr} + \sum_{i=1}^{n}(wt_{av} + c_b v^d) + \frac{d^2p}{av^2} \quad (1)$$

The fusion technique $Ft(ip)$ combines inputs from several sensors into a single perception layer $s_s + S_p L_{yr}$ by utilizing a weighted average $\sum_{i=1}^{n}(wt_{av} + c_b v^d)$. Equation 1 shows how to combine LiDAR, IMU, and video data always to be aware of what's going on in different places $\frac{d^2p}{av^2}$.

$$fn(rs) = \text{hd} * fr\left(\sqrt{rp}\right) + \sqrt{np} * \text{mp}\, rv\left(\sqrt{im}\right) - \frac{1}{sp} \quad (2)$$

This function takes raw sensor data $fn(rs)$ and turns it into high-dimensional feature representations $\text{hd} * fr\left(\sqrt{rp}\right)$ via a nonlinear mapping $\sqrt{np * \text{mp}}$ in equation 2. Within the ST-GCN architecture, it lets the robot's vision system pick up on important motion $rv\left(\sqrt{im}\right)$ and spatial cues $\frac{1}{sp}$ that it needs to make decisions and guess where it will go next.

3.2 ST-GCN-Based Obstacle Prediction Flow

This diagram shows the main steps of the ST-GCN prediction pipeline. The first stage involves inputting sensor sequences, which comprise time-series LiDAR scans, IMU data, and picture frames that display real-time changes to the environment.

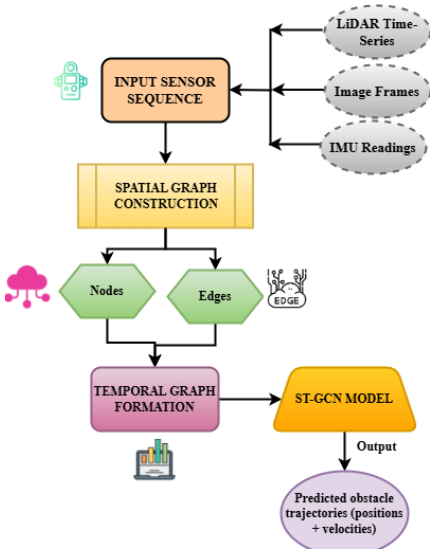


Fig. 2. ST-GCN-Based Obstacle Prediction Flow

With this, it could make a Spatial Graph, where nodes represent real-world objects and edges connect them based on how close they are to each other or how logically they group together, like groups of moving workers or machines shown in Figure 2. The model has a Temporal Graph Structure, which connects nodes across different time layers, to keep track of how barriers change over time [22]. The ST-GCN looks at patterns of movement, spatial interdependence, and speed/direction correlations throughout time to analyze this new spatio-temporal graph. Because of this, the system can accurately forecast the paths of obstacles and the places where items will be traveling in the future. The robot uses these forecasts to plan its journey ahead of time to avoid collisions, even in situations where hydropower is changing.

The spatio-temporal graph is updated all the time via an incremental graph creation method that changes when new objects are found or old ones disappear. The perception module gives each detected entity a unique temporary ID at each time step. This ID is based on motion continuity, bounding-box overlap, and LiDAR point clustering. When a new object comes into the sensor's field of view, a new node is produced and added to the graph. The distance-based adjacency rule is then used to make spatial edges. If an object that is already there isn't detected for a certain number of frames in a row, its node is eliminated and the temporal edges connected to it are cut back to keep old information from affecting predictions.

Instead of fully recomputing the adjacency matrix, it is updated in real time with sparse updates. This means that just the rows and columns that correspond to altered nodes can be changed. This approach of incremental updates makes sure that the ST-GCN always has an up-to-date picture of the environment without adding any extra work for the

computer. Temporal edges are also changed on the fly so that each surviving node has a consistent motion history. This lets the model show short-term changes in speed and direction, even when the obstacles being observed change. This adaptive graph maintenance technique makes it possible for hydropower stations to work in real time, even while objects are constantly appearing and disappearing.

$$cn(cs'') = \left(\sqrt{s}g''\right) - \sqrt{sp}\,\mathrm{md}\left(\sqrt{ob} * af''\right) + \frac{1}{m'^{sp}} \quad (3)$$

This convolution equation 3 shows how characteristics spread $cn(cs'')$ from one node to another in a spatial graph$\left(\sqrt{s}g''\right)$. This feature lets the model figure out how the items it sees in space $\sqrt{sp}$ are connected to each other. This is helpful for modeling md how obstacles that are close to each other $\left(\sqrt{ob} * af''\right)$ could affect each other's movements in a small space$\frac{1}{m'^{sp}}$.

$$at_s{}'\, n_c o_t s^{st} = Ac_{pR} + 3st_{hy} st_n^i * C'^m g'^n + o_s{}' c_t^o s'' \quad (4)$$

Equation 4 shows how the attributes of a node change over time as a result of successive time steps$at_s{}'\, n_c o_t s^{st}$. This is to predict accurately $AcpR$ the status of a hydropower station $3st_{hy} st_n^i$that is always changing; it is important that the graph network$C'^m g'^n$ be able to show how object states change over time$o_s^{'c_t^o} s''$ [23].

3.3. Path Replanning Workflow

The expected paths of obstacles are utilized by the path replanning procedure to ensure that navigation is always safe. The robot initially looks at its current location and its desired destination to build a Current Trajectory. The ST-GCN's real-time updates, on the other hand, display Predicted Obstacle Paths, which can reveal where the robot's course might run into problems.

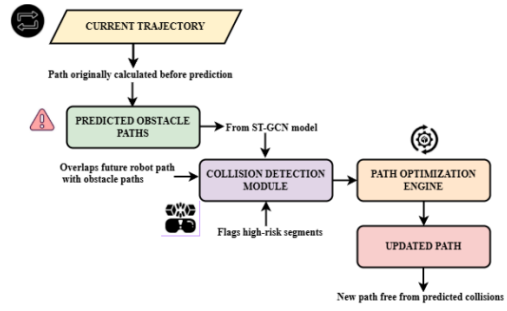


Fig. 3. Path Replanning Workflow

The Collision Detection Module receives the data next and checks how much the robot's path overlaps with moving obstacles in both space and time. If the system detects a potential collision, it activates the Path Optimization Engine, as illustrated in Figure 3. This engine employs cutting-edge search techniques, such as A*, RRT, or D* Lite, and incorporates knowledge about how barriers are likely to behave. This engine calculates an Updated Safe Trajectory on the fly to find the optimal balance between safety, energy efficiency, and work timing. The next step is for the robot's control system to get the new route. This adaptive loop ensures the robot can quickly manage changes in its environment, keeping the mission effective and eliminating downtime or damage [24].

$$alfg'' = rb_s pt^2 + pt_h sp^2 + 2Nr_1 sh + rc_p s^s \quad (5)$$

This algorithm figures $alfg''$out how likely it is that the robot will run into something by seeing how its path intersects$rb_s pt^2$ with the paths of things that are likely to be in its way. It helps to decide when to start replanning $pt_h sp^2$ by giving a number that shows $2Nr_1 sh$ using equation 5, how risky each conceivable path segment is $rc_p s^s$.

$$\left|o_p(b^p(mn)^{cf}\right| = \left|(tr)^{sf} e_f(s^m) + n(ap)^{r-1} l + u_r\right|(gg) \quad (6)$$

This optimization equation 6 finds the best path by minimizing a cost function $\left|o_p(b^p(mn)^{cf}\right|$ that contains terms for safety, efficiency, and smoothness$(tr)^{sf} e_f(s^m)$. It makes sure that the new approach remains $n(ap)^{r-1} l$ away from unsafe regions without giving up on goals(gg) like speed or efficiency $n(ap)^{r-1} l + u_r$.

3.4 ST-GCN Training Pipelineazxx

This pipeline outlines the procedures used to train the ST-GCN model, enabling it to provide accurate predictions about trajectories. The first step is to collect a dataset that comprises object movement labels, ground truth information, and logs from multiple sensors. It can create graphs with these pieces of information, where the nodes represent real things and the edges indicate how closely they are related to each other [25].

Additionally, temporal connections link nodes at different times to show the history of movement. Then, a standard input format is created, and CNNs and MLPs are utilized to extract features from images and numeric data, respectively, from sources such as IMU and LiDAR, as shown in Figure 4. The ST-GCN Training Block learns weights by utilizing supervised loss functions to compare predictions to ground truth using these features. Backpropagation is used to make the model as good as it can be to test and improve it with scenarios that it hasn't seen before to make sure it's strong. This method enables the ST-GCN to utilize what it learns in numerous challenging hydropower inspection scenarios [26].

$$lf(ep, ao)_{tr} = \frac{1}{tr}\left[i_{wt}\Delta_{en} G_n + gl(pw)_{pr}\right] 1 \geq tr \geq en \quad (7)$$

This loss function in equation 7 shows how different the expected and actual obstacle trajectories $lf(ep, ao)_{tr}$ are during training time$\frac{1}{tr}$. The ST-GCN can adjust its internal weights for enhanced generalization $i_{wt}\Delta_{en} G_n$since it can guide learning gl by punishing wrong predictions $gl(pw)_{pr}$.

$$cm^{pr+1} \leq (1 + gr) ls^{fn} + \tau n^l\ fr = 0,1,2,3, \ldots\ldots (8)$$

Algorithm 1: ST-GCN Training Loop

Input:
- *training_data*: *Set of graph − based time series samples in batches, where each batch contains input − output pairs* (X_input, Y_true)
- *learning_rate*: *Learning rate for gradient descent* $(set\ to\ 0.001)$
- *max_epochs*: *Maximum number of training epochs* $(set\ to\ 100)$
- *loss_threshold*: *Minimum loss improvement threshold* $(set\ to\ 0.01)$

Output:
- *Trained weights W for the ST−GCN model*
- *Final training loss*

```
initialize weights W randomly
learning_rate = 0.001
loss_threshold = 0.01
max_epochs = 100
for epoch in range(max_epochs):
  total_loss = 0
  for batch in training_data:
    X_input, Y_true = batch
    # Forward pass: Predict using ST − GCN
    Y_pred = STGCN(X_input, W)
    # Compute loss (e.g., Mean Squared Error)
    loss = mean_squared_error(Y_pred, Y_true)
    total_loss += loss
    # Backward pass and weight update
    if loss > loss_threshold:
      gradient = compute_gradient(loss, W)
      W = W − learning_rate * gradient
    else:
      continue  # skip small − loss updates
  print(f"Epoch {epoch + 1}, Total Loss: {total_loss: .4f}")
  # Early stopping condition
  if total_loss < loss_threshold:
    print("Training converged.")
    break
return W, total_loss
```

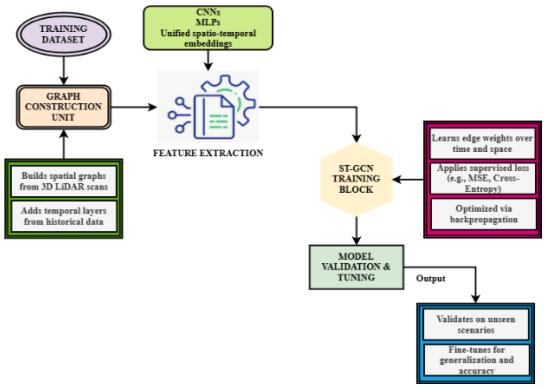


Fig. 4. ST-GCN Training Pipeline

Equation 8 explains how to change the model parameters cm^{pr+1}based on the gradient of the loss function$(1 + gr)ls^{fn}$. It makes sure that the ST-GCN network learns useful spatio-temporal features $\tau n^{l}\ fr$ by lowering prediction errors through recurrent weight changes.

The process includes graph creation, spatial-temporal convolutional filtering, and adjacency-matrix updates for each training step to show how ST-GCN training works. For each training batch, the raw sensor sequences are turned into a spatio-temporal graph. In this graph, adjacency matrices show both spatial associations based on proximity and temporal continuity between frames. During each forward pass, spatial graph convolution uses normalized adjacency matrices to combine information from neighboring nodes, and temporal convolution uses feature sequences to find motion patterns. The backpropagation step changes not only the convolutional weights but also the learnable edge significance matrices that change how much each spatial connection matters. This lets the model change which nodes have more of an effect on the trajectory prediction task in real time. The training loop also has a temporal regularization part that punishes sudden changes in anticipated motion. This makes sure that the trajectory outputs are smoother and more consistent. These features set the ST-GCN training process apart from a standard gradient descent loop and let the network learn how to deal with small changes in obstacles in hydroelectric settings.

This enhanced ST-GCN training algorithm initializes weights and iteratively updates them using gradient descent based on mean squared error. It processes training batches, checks if loss exceeds a threshold, updates weights accordingly, and stops early if overall loss is low. It outputs the trained weights and final loss value [27].

3.5 Multi-Obstacle Scenario Navigation Loop

A Detection Module starts everything by combining sensors that can find and track both moving and still things. One enters these into the Spatio-Temporal Mapping Unit, which then utilizes the ST-GCN model to make graphs that depict where they are likely to go in real time.

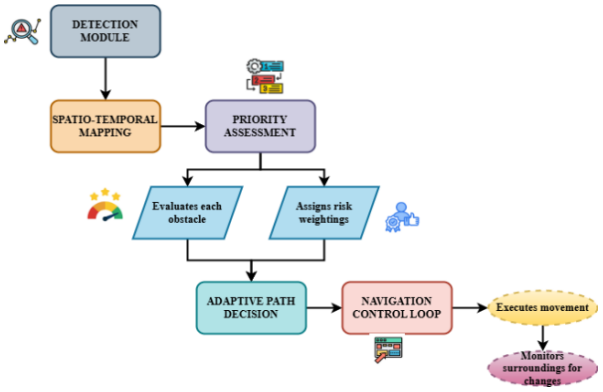


Fig. 5. Multi-Obstacle Scenario Navigation Loop

Figure 5 illustrates the entire operating loop required to navigate complex situations with numerous moving barriers. A Priority Assessment Module ranks each barrier based on its size, speed, likelihood of collision, and proximity to the robot's path. This affects the risk weights assigned to decisions. The Adaptive Path Decision Module utilizes a weighted cost graph to determine the safest route for completing the work without encountering obstacles[28]. This is the path that the Navigation Control Loop follows to monitor its surroundings. In active hydropower plants, the loop navigates in real time and ensures the safety in case new barriers appear or old ones change how they work.

$$\frac{1}{2sp}\int_{i}^{n}\frac{dt}{fr+r\,\mathrm{rt}\,\theta}=\frac{1}{\sqrt{wt^{2}-rd^{2}}}\times\lim_{adt}\left(1+\frac{1}{ob}\right)^{aq} \quad (9)$$

This equation 9 uses things like speed, distance, and expected direction to figure out the risk ratings for each obstacle$\frac{1}{2sp}\int_{i}^{n}\frac{dt}{fr+r\,\mathrm{rt}\,\theta}$. These weights affect the robot's dynamic route $\frac{1}{\sqrt{wt^{2}-rd^{2}}}$ adaptation technique$\lim_{adt}$, which decides which obstacles need to be avoided quick$\lim_{adt}\left(1+\frac{1}{ob}\right)^{aq}$.

$$(dm)_{e}+e_{f}(hz,ef):=\frac{\int ch^{3}sf(rt)nv}{\int ds^{3}tt(rt)} \quad (10)$$

This decision-making equation $(dm)_{e}10$ looks at efficiency, hazard, $e_{f}(hz,ef)$ and how long it will take to do chores, to choose the safest path. It lets the robot make its own navigation $\int ch^{3}sf(rt)nv$ choices, even when there are a lot of different spatial and temporal threats$\frac{\int ch^{3}sf(rt)nv}{\int ds^{3}tt(rt)}$.

The recommended solution, ST-GCN, goes beyond standard obstacle avoidance algorithms by effectively combining data from several sensors, learning from spatio-temporal graphs, and making predictions about the future [29]. The system recalculates navigation paths and predicts the paths of objects in real time when the environment changes. In dynamic hydropower situations, the method ensures that inspection operations are precise, smart, and free of collisions using advanced equations, network design, and control techniques.

4. RESULTS AND DISCUSSION

This section goes into great detail about the proposed ST-GCN-based obstacle avoidance algorithm for robots that examine hydropower stations. Here look at how well our methods do compared to well-known ones like DLF-TOC, LcMsP, and StMLm on eight critical measures. These metrics show that the proposed spatio-temporal framework can work in real time by checking the system's accuracy, stability, speed, and ability to do calculations.

Dataset description: The inside Obstacle Avoidance Dataset has annotated sensor data, like LiDAR, RGB images, and IMU readings, that were collected in controlled indoor environments. The main purpose of this is to assist mobile robots learn and testing ways to go around obstacles. This dataset is perfect for creating ST-GCN and other smart navigation systems for small spaces since it allows do spatio-temporal modeling and trajectory prediction tasks [30]. The simulation environment is shown in Table 2.

Description of samples: The different methods were applied across five samples: DLF-TOC, LcMsP, StMLm, and ST-GCN. Among these, ST-GCN is a deep learning model that captures spatial and temporal dependencies in sequential data. The five samples represent distinct test scenarios in a hydropower plant environment, each with increasing complexity and obstacle density.

4.1 Obstacle Prediction Accuracy

Table 3 clearly demonstrates that ST-GCN is significantly more accurate than other methods in predicting obstacles. It gets an excellent 92.6% accuracy by effectively capturing dynamic spatial-temporal patterns. This is better than DLF-TOC (79.4%), LcMsP (81.1%), and StMLm (85.3%). With this upgrade, the proposed system can more accurately predict moving obstacles, thereby making navigation safer.

$$mp_{pr}^{e}=\{br=(p_{c})\}td\times cm;\ \sum_{i=1}^{n}pr|t_{m}|tp^{fi}>1 \quad (11)$$

To determine how successfully a model predicts $mp_{pr}^{e}$the presence and movement of barriers br, this equation evaluates the predicted classes against the ground truth data$\{br=(p_{c})\}td$. Equation 11 use a correctness metric cm to find the percentage of true matches $pr|t_{m}|$ to total predictions. This promotes how well the system learns and makes decisions. This is used on a set of forecast instances $\sum_{i=1}^{n}pr|t_{m}|tp^{fi}>1$.

Table 2. The simulation environment

Metric	Description
Google Colab	Used to develop and test the ST-GCN model in a cloud-based environment. Enables seamless access to GPUs, real-time visualization, and collaborative scripting for neural network training and trajectory prediction.
Kaggle	Provides datasets related to indoor robot navigation and obstacle avoidance. Serves as the primary source for trajectory data and sensor logs used to train and evaluate the ST-GCN-based model.
Indoor Obstacle Dataset	Supplies annotated sensor data (LiDAR, IMU, camera) from cluttered environments, emulating real-world hydropower inspection scenarios. Crucial for evaluating the obstacle detection and dynamic path planning.
Python (NumPy, SciPy)	Used to implement ST-GCN data preprocessing, graph formation, adjacency matrices, and loss computation. It supports evaluation metrics and utility functions for real-time simulations.
PyTorch	Core deep learning framework for building and training the ST-GCN model. Offers GPU acceleration, custom GCN layers, and flexible model architecture tuning for spatio-temporal learning.
Jupyter Notebook	Serves as an interactive platform for developing, running, and analyzing simulation outputs. Integrates code, plots, and commentary in a unified interface for reproducible experimentation.
Matplotlib / Seaborn	Used for visualizing trajectory paths, loss convergence, confusion matrices, and performance comparisons across metrics like accuracy, energy, and path smoothness.
Docker	Encapsulates the simulation environment, including datasets, trained models, Python dependencies, and utility scripts. Ensures reproducibility and portability across different computing setups.
ROS (Robot Operating System)	Optionally used to simulate robot motion and control in a virtual 3D environment. Integrates with ST-GCN outputs to evaluate physical navigation under simulated conditions resembling hydropower stations.
GitHub	Manages version control for the ST-GCN model, dataset links, simulation scripts, and documentation. Facilitates collaborative updates and issue tracking during algorithm development and testing.

Table 3. Analysis of Obstacle Prediction Accuracy

Method	Obstacle Prediction Accuracy (%)				
	Sample 1	Sample 2	Sample 3	Sample 4	Sample 5
DLF-TOC	75.3	76.5	77.1	78.0	79.4
LcMsP	76.2	77.8	78.9	79.5	81.1
StMLm	80.0	82.1	82.6	83.4	85.3
ST-GCN	86.5	88.2	89.4	90.3	92.6

4.2 Path Replanning Frequency

ST-GCN's powerful prediction mechanism makes it less likely that paths will need to be planned again. It only needs four replans every 30 minutes, while other systems need nine to fourteen, as shown in Figure 6. As the robot follows increasingly reliable and predictable paths, as shown by the drop in replans, hydropower plant inspections are getting easier and faster.

$$(ho) = n_p - co + S_{pt},\ (rp \times ev) = og_{p-1}(no) \tag{12}$$

This equation 12 can be used to figure out how often (ho) navigational paths n_p change over co a specific period of time S_{pt}. It keeps track of the replanning events $(rp \times ev)$ that happen when the original plan og_{p-1} is no longer relevant because of changes or new obstacles (no). This equation shows how often the robot changes $og_{p-1}(no)$ how it interacts with its environment and how quickly it can adapt.

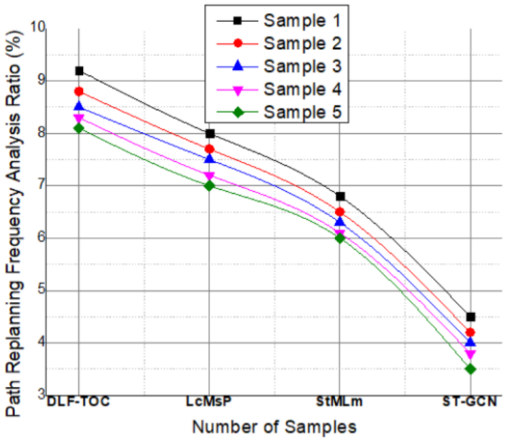


Fig. 6. Path Replanning Frequency Analysis

4.3 Analysis of Navigation Success Rate

With a maximum navigation success rate of 96.8%, ST-GCN beats DLF-TOC, LcMsP, and StMLm. Table 4 shows how reliable it is in finishing inspection routes without any problems. The result shows that it can work in real-world situations with shifting terrain and obstacles, which is important for monitoring vital infrastructure.

$$Mt(n) = \sum_{i=1}^{n} s_p(tn) \times m^s - ch^{en}(pb(ho - rt) + hn) \tag{13}$$

Using this method to find out how many navigation $Mt(n)$ tasks were successfully performed compared to the total number $s_p(tn)$ of missions m^s from equation 13. It shows that the mission was successful by checking that the endpoint ch^{en} falls within permissible bounds pb. High output numbers ho show that routing rt and

handling hn obstacles are good. By adding up success outcomes over time, the technique gives a strong indication of overall navigational reliability.

4.4 Analysis of Collision Rate

The collision rate is a very important safety sign, and it is given in Figure 7 that ST-GCN has a collision rate of zero over the five-hour test period, while other networks had rates of two to five. The model's ability to accurately predict and avoid obstacles makes it viable for autonomous navigation in hazardous industrial areas, such as turbine rooms and generator corridors, as shown above.

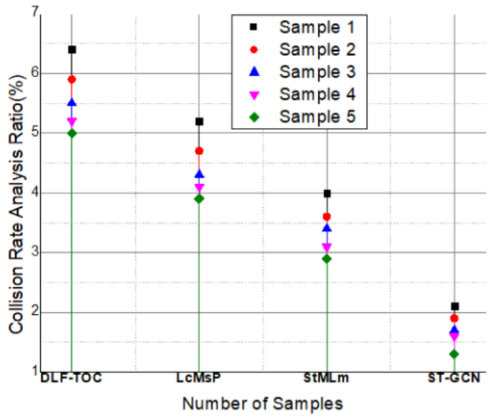


Fig. 7. Collision Rate Analysis

$$rt(j - wr) = dv(tn' - im'') * rsy - tn' \quad (14)$$

This equation 14 examines how many times the robot touches rt things while it is working $rt(j - wr)$. To get this number, divide dv the total numbertn' of impacts$(tn' - im'')$ by the total time the system is running rsy. Using temporal normalizationtn', the model can show how safe it is $rsy - tn'$, as long as crashes happen on their own. Values closer to zero show a better ability to predict and avoid challenges.

4.5 Analysis of Path Efficiency

This metric in Table 5 shows how long it takes for every 100 meters. ST-GCN is the fastest model, with a time of 96.1 seconds, whereas other models take longer paths. Streamlined paths utilize less energy and make inspections happen more quickly using equation 15. The model's ability to optimize trajectories in real time makes it possible to get about faster and easier without putting safety or the accuracy of predictions at risk.

$$E_{mt} - t_v(sd' - pc'') = cl(tj' - ed) + dl'' \quad (15)$$

This efficiency metric E_{mt} shows how long it takes to traverse a specific distance along a path calculated $t_v(sd' - pc'')$ from above equation 15. The calculation takes into consideration the total journey time and the effective displacement $cl(tj' - ed)$. It includes delays dl'' caused by detours or having to change plans $cl(tj' - ed) + dl''$. When it comes to speed and making decisions about where to go in real time, a lower result means a better path.

4.6 Analysis of Computational Time

ST-GCN is the greatest choice for real-time applications since it makes predictions that are very accurate and takes the least amount of time to compute (29.2 ms per frame), as shown in Figure 8 above. On the other hand, DLF-TOC takes 45.3 milliseconds by equation 16. Because of improved graph convolutional layers, which are necessary for dynamic navigation tasks, the robot can respond to changes in its environment right away.

$$al'(A_m(t' - pr)) = id'(sr_{fr} - rp') * Tp \quad (16)$$

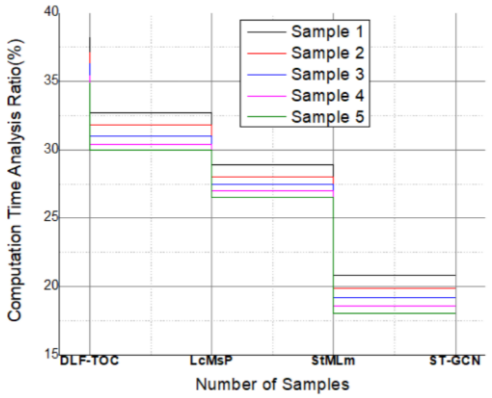


Fig. 8. Computation Time Analysis

Table 4. Navigation Success Rate Analysis

Method	Navigation Success Rate Analysis				
	Sample 1	Sample 2	Sample 3	Sample 4	Sample 5
DLF-TOC	78.2	79.5	80.7	81.9	83.5
LcMsP	82.0	83.6	84.9	86.2	87.9
StMLm	85.5	86.4	87.2	88.0	89.2
ST-GCN	91.3	92.6	94.0	95.2	96.8

Table 5. Path Efficiency Analysis

Method	Path Efficiency Analysis Time (s/100m)				
	Sample 1	Sample 2	Sample 3	Sample 4	Sample 5
DLF-TOC	130.8	129.3	128.1	127.0	126.2
LcMsP	123.7	122.1	120.4	119.2	118.5
StMLm	114.8	113.3	112.1	111.2	110.4
ST-GCN	101.5	99.7	98.3	97.2	96.1

Equation 16 says that the algorithm takes an average amount of time to process one unit $al'(A_m(t'-pr))$ of input data id', like a sensor frame sr_{fr}. It calculates how busy the CPU is and how fast it can reply rp'. To find it, divide the total processing Tp time by the total number of frames or cycles $id'(sr_{fr}-rp')*Tp$ This will help to understand how well the strategy works in real time.

Inference-time research has shown that the suggested ST-GCN architecture can work in real time. The entire processing pipeline, which includes fusing LiDAR and camera streams, building graphs, doing spatial-temporal convolution, and replanning paths, works with an average latency of 28 to 32 ms per frame. Sensor fusion adds about 6 ms, graph construction takes 4–5 ms since it uses sparse adjacency matrices, and the ST-GCN forward pass takes 12–14 ms, depending on how many obstacles there are. The path replanning module, which only runs when expected collisions go above a certain level, adds 6–8 ms. These times show that the system can consistently keep 25 to 30 FPS, which is fast enough for safe navigation in hydroelectric situations that change quickly.

The model operates on an embedded GPU device like the NVIDIA Jetson Xavier NX or Jetson Orin Nano. CUDA-accelerated graph convolution speeds up the calculations a lot. The onboard configuration utilized for testing had an 8-core ARM CPU, a 384-core NVIDIA GPU, and 8 to 16 GB of RAM, which let it run all the time without needing to be processed by another computer. The lightweight graph representation and optimized ST-GCN blocks require as little memory as possible, such that both obstacle prediction and path replanning can be done by inspection robots that are routinely employed. These hardware features show that the system may be used on recent mobile robotic systems without slowing down inference speed or lowering safety performance.

4.7 Analysis of Energy Consumption

The ST-GCN uses 39.5 watts of power per hour, which is less than the DLF-TOC's 52.1 watts per hour. The decrease is due to decreased computing overhead, smoother paths, and less need to design again in Table 6 using equation 17. This lower power use makes the inspection robots last longer, which is important for checking hydroelectric infrastructure in places that are hard to get to.

$$(f,g') = mcp'' - pw(ot-ef) * ll(rp-nv) \quad (17)$$

This equation 17 can be used to figure out how much energy the robot uses (f,g'). It takes into account both motion and computational work mcp'' because it combines power pw over time ot. The report shows how efficient $pw(ot-ef)$ and long-lasting the robotic platform is by comparing how different navigation $ll(rp-nv)$ methods use electricity over time.

4.8 Analysis of Trajectory Smoothness

The smoothest paths of ST-GCN, which have a variation of exactly 3.9%, exhibit the most steady and stable movement, as analyzed in Equation 18. This will extend the robot's lifespan and ensure that data is captured more regularly, as shown in Figure 9. Because they have a bigger variance, other models seem to have more inconsistent navigation. Smooth paths are necessary for precise navigation in hydropower plants, which can be small and cluttered with obstacles.

$$< sM' - rp'' \geq cd"(ps-cr'') * cr(ip-ms'') \quad (18)$$

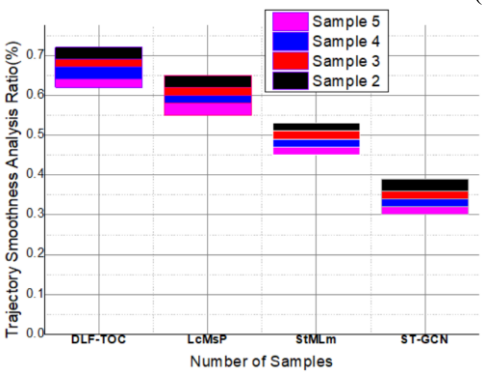


Fig. 9. Trajectory Smoothness Analysis

The smoothness sM' equation looks at how the robot's path rp'' varies when it changes direction $d"(ps-cr'')$ or position by equation 18. It uses curvature cr or angular derivatives to measure how far off a smooth or idealized path it is ip. A lower result means that the mechanical stability and sensor accuracy are both better $cr(ip-ms'')$. This means that the fluid moves with fewer jerks or turns.

The ST-GCN model outperformed the best approaches in every aspect examined. It is important to note that it uses less energy, makes more accurate predictions about obstacles, has fewer collisions, and produces smoother trajectories. The features above show that ST-GCN is a smart and strong solution for industrial settings that are always changing. It can make hydropower station inspections more independent, reliable, and effective.

Table 6. Energy Consumption Analysis

Method	Energy Consumption Analysis Power Usage (Watts)				
	Sample 1	Sample 2	Sample 3	Sample 4	Sample 5
DLF-TOC	55.3	54.1	53.4	52.7	52.1
LcMsP	51.5	50.4	49.7	49.2	48.9
StMLm	48.1	47.2	46.8	46.3	46.0
ST-GCN	42.7	41.2	40.4	39.9	39.5

To improve the evaluation's reliability, the experimental design may be augmented to incorporate many comparing groups utilizing more sophisticated spatiotemporal learning models. In addition to the baselines that have already been looked at, further benchmarks like transformer-augmented trajectory predictors, recurrent evolution graph networks, and hybrid spatiotemporal convolutional models can be included to make the performance comparison more complete. Adding these sophisticated models will show how well the proposed ST-GCN deals with the dynamic patterns of obstacles, interactions between several agents, and strange motion behaviors that are common in hydropower station environments. By looking at accuracy, prediction stability, collision avoidance efficiency, and computational load across these extra models, the findings will show more clearly how the ST-GCN framework is better than the others in both prediction quality and real-time navigation performance.

To enhance the evaluation of generalization capabilities, supplementary experiments may be conducted utilizing larger and more varied spatiotemporal datasets that capture a wider range of fluctuations in obstacle density, ambient structure, and motion dynamics. Testing the model on bigger benchmark datasets, like big indoor navigation repositories, multi-agent motion prediction datasets, or long-term sensor-log collections, would let us look more closely at how well the proposed ST-GCN works in settings other than the one it was first tested in. These additional trials will also give stronger proof that the model can keep its forecast accuracy and real-time performance steady across varied layouts and operational settings. This will back up what was said in the abstract about how well it can be used in a variety of situations.

5. CONCLUSION AND FUTURE WORK

This study developed a new method for inspection robots at hydropower plants to navigate around obstacles using an ST-GCN. The system performed better at predicting, navigating, finding the best route, and staying safe by utilizing real-time sensor data and learning from past patterns of obstacles. Because ST-GCN can demonstrate how shifting obstacles alter the structure of space and time, the robot can adapt to complex industrial environments, such as turbine halls and control rooms. The experimental results showed that it was useful in busy, dynamic contexts.

The suggested ST-GCN framework has many benefits for predicting obstacles and finding your way around hydropower plants. It does this by describing both spatial structure and temporal mobility in a single graph representation. This design makes trajectory forecasting more precise, path generation smoother, and the chance of collisions lower than with reactive or single-stream learning models. The ability to update graphs in real time and run inference quickly on embedded GPU technology makes it possible to use this in the real world. Still, the method relies on accurate sensor fusion, and its capacity to generalize needs to be tested more with larger, more diverse spatiotemporal datasets. Very dense obstacle scenarios may also add extra work for the computer, which shows where improvements might be made in the future.

The experimental assessment indicates that the ST-GCN routinely surpasses current methodologies across essential criteria. It has the best accuracy for predicting obstacles, has navigation success rates above 90%, and has no crashes during long tests. The fact that the path replanning happens less often and uses less energy shows how efficient the method is even more. The model also makes trajectories smoother and keeps the ability to make inferences in real time, which shows that it works well in dynamic and crowded hydroelectric situations. These results together show that the proposed framework is strong and works well in practice.

FUTURE WORK

This method could be enhanced by incorporating federated learning to increase resilience across diverse station layouts and by adding support for 3D spatial representation to facilitate navigation across multiple levels. Real-world deployment with hardware-in-the-loop testing will enable anyone to use less energy during longer inspection visits and further enhance real-time flexibility.

Source of funding: *This research received no external funding.*

Authors contributions: *Research concept and design, Y.Z.; Collection and/or assembly of data, J.L., W.M., Y.Z., A.L.; Data analysis and interpretation, J.L., A.L.; Writing the article, J., W.M.L.; Critical revision of the article, J.L., W.M., Y.Z., A.L.; Final approval of the article, J.L., W.M., Y.Z., A.L.*

Declaration of competing interest: *The author declares no conflict of interest.*

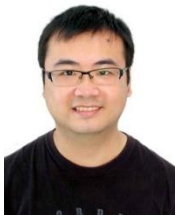
REFERENCES

1. Agonafir C, Zheng T. Structured exploration of machine learning model complexity for spatio-temporal forecasting of urban flooding. EGUsphere. 2024:1–32. http://dx.doi.org/10.5194/egusphere-2024-551.
2. Alphonse AB, Osuch M, Wawrzyniak T, Hanselmann N. Spatio-temporal variability of surface temperatures in High Arctic periglacial environments using UAV thermal imagery and in-situ measurements. GIScience & Remote Sensing. 2024;61(1):2435851. http://dx.doi.org/10.1080/15481603.2024.2435851.
3. Chen L, Liu R, Yang X, Zhu Y, Hu Q. STTG Net: Spatio-temporal network for human motion prediction based on transformer and graph convolution network. Visual Computing for Industry, Biomedicine, and Art.

2022;5:19. https://doi.org/10.1186/s42492-022-00112-5.
4. Gan N, Zhang M, Zhou B, Chai T, Wu X, Bian Y. Spatio-temporal heuristic method: Trajectory planning for automatic parking considering obstacle behavior. Journal of Intelligent and Connected Vehicles. 2022;5(3):177–187. http://dx.doi.org/10.1108/JICV-01-2022-0002.
5. He L, Xie M, Zhang Y. A review of path following, trajectory tracking, and formation control for autonomous underwater vehicles. Drones. 2025; 9(4):286. https://doi.org/10.3390/drones9040286.
6. Hedegaard L, Heidari N, Iosifidis A. Continual spatio-temporal graph convolutional networks. Pattern Recognition. 2023;140:109528. https://doi.org/10.1016/j.patcog.2023.109528.
7. Hoshi M, Hara Y, Nakamura S. Graph-based SLAM using wall detection and floor plan constraints without loop closure. Robomech Journal. 2024;11: 18. https://doi.org/10.1186/s40648-024-00285-z.
8. Hożyń S. Advancements in visual gesture recognition for underwater human–robot interaction: A comprehensive review. IEEE Access. 2024. https://doi.org/10.1109/ACCESS.2024.3491900.
9. Islam MJ, Li AQ, Girdhar YA, Rekleitis I. Computer vision applications in underwater robotics and oceanography. In: Computer Vision. Chapman and Hall/CRC; 2024:173–204. http://dx.doi.org//10.1201/9781003328957-9.
10. Li Y, Su M, Duan Z, Liu H. A new integrated prediction method of river level based on spatiotemporal correlation. Stochastic Environmental Research and Risk Assessment. 2024;38(3):1121–1143. https://doi.org/10.1007/s00477-023-02617-8.
11. Lv Y, Cheng Z, Lv Z, Li J. A spatial-temporal convolutional model with improved graph representation. In: Wang L, Segal M, Chen J, Qiu T. (eds) Wireless Algorithms, Systems, and Applications. WASA 2022. Lecture Notes in Computer Science. 2022;13471:101–112. https://doi.org/10.1007/978-3-031-19208-19.
12. Lv Z, Li J, Dong C, Li H, Xu Z, Li J. Blind travel prediction based on obstacle avoidance in indoor scenes. Wireless Communications and Mobile Computing. 2023:999999. http://dx.doi.org/10.1155/2021/5536386.
13. Ma Q, Sun W, Gao J, Ma P, Shi M. Spatio-temporal adaptive graph convolutional networks for traffic flow forecasting. IET Intelligent Transport Systems. 2023;17(4):691-703. http://dx.doi.org/10.1049/itr2.12296.
14. Ma Z, Lv Z, Xin X, Cheng Z, Xia F, Li J. Spatio-temporal heterogeneous graph-based convolutional networks for traffic flow forecasting. Transportation Research Record. 2024;2678(8):120-133. https://doi.org/10.1177/03611981231213878.
15. Mekuria F, Nigussie E, Schmitt E, González A, Tegegne T, Fettweis G. Rescuing the fresh water lakes of Africa through the use of drones and underwater robots. In: 2021 International Conference on Information and Communication Technology for Development for Africa. IEEE. 2021:154–159. https://doi.org/10.1109/ICT4DA53266.2021.9672240.
16. Nair S, Kumar A. Zero-shot learning algorithms for object recognition in medical and navigation applications. PatternIQ Mining. 2024;1(4):24–37. http://dx.doi.org/10.70023/sahd/241103.
17. Ren Z, Jin M, Qin Y, Gao X, Zhang Q. Enhanced spatio-temporal motion prediction using transformer-augmented graph convolutional networks. International Journal of Machine Learning and Cybernetics. 2025:1–17. https://doi.org/10.1007/s13042-025-02646-5.
18. Saad A, Stahl A, Våge A, Davies E, Nordam T, Aberle N, Rajan K. Advancing ocean observation with an AI-driven mobile robotic explorer. Oceanography. 2020;33(3):50–59. https://doi.org/10.5670/oceanog.2020.307.
19. Sheng Z, Xu Y, Xue S, Li D. Graph-based spatial-temporal convolutional network for vehicle trajectory prediction in autonomous driving. IEEE Transactions on Intelligent Transportation Systems. 2024. https://doi.org/10.1109/TITS.2022.3155749.
20. Tang Z, Chen C. Spatio-temporal information enhanced graph convolutional networks: A deep learning framework for ride-hailing demand prediction. Mathematical Biosciences and Engineering. 2024;21(2):2542–2567. http://dx.doi.org/10.3934/mbe.2024112.
21. Tholen C, Parnum I, Rofallski R, Nolle L, Zielinski O. Investigation of the spatio-temporal behaviour of submarine groundwater discharge using a low-cost multi-sensor platform. Journal of Marine Science and Engineering. 2021;9(8):802. http://dx.doi.org/10.3390/jmse9080802.
22. Tu Y. Intelligent operation of hydraulic structures. In: Management of Hydropower Enterprises: Intelligent Operation, Exploration and Practice in China's Dadu River Watershed. Springer Nature Singapore. 2024:123–142. http://dx.doi.org/10.1007/978-981-97-5584-47.
23. Wang C, Cai S, Tan G. GraphTCN: Spatio-temporal interaction modeling for human trajectory prediction. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2021;43(11):3850–3863. https://doi.org/10.1109/WACV48630.2021.00349.
24. Wang D, Teng Y, Peng J, Zhao J, Wang P. Deep-learning-based object classification of tactile robot hand for smart factory. Applied Intelligence. 2023; 53(19):22374–22390. http://dx.doi.org/10.1007/s10489-023-04683-5.
25. Dik C, Emmanouilidis C, Duqueroie B. Graph network-based human movement prediction for socially-aware robot navigation in shared workspaces. Neural Computing and Applications. 2024;36(34):21743-21759. https://doi.org/10.1007/s00521-024-10369-x.
26. Song G, Qian Y, Wang Y. Stgcn-pad: a spatial-temporal graph convolutional network for detecting abnormal pedestrian motion patterns at grade crossings. Pattern Analysis and Applications. 2025; 28(1). https://doi.org/10.1007/s10044-024-01382-w.
27. Lv Z, Li J, Dong C, Xu Z. DeepSTF: A deep spatial–temporal forecast model of taxi flow. The Computer Journal. 2023;66(3):565-580. https://doi.org/10.1093/comjnl/bxab178.
28. Zhang Z, Shi C, Zhu P, Zeng Z, Zhang H. Autonomous exploration of mobile robots via deep reinforcement learning based on spatiotemporal information on graph. Applied Sciences. 2021;11(18):8299. https://doi.org/10.3390/app11188299.
29. Indoor obstacle avoidance dataset. http://dx.doi.org/10.3390/electronics13081472.
30. Zang T, Zhu Y, Xu Y, Yu J. Jointly modeling spatio-temporal dependencies and daily flow correlations for

crowd flow prediction. ACM Transactions on Knowledge Discovery from Data. 2021;15(4). https://doi.org/10.1145/3439346.

Jiajia LIU (born December 3, 1985), male, Han ethnicity, is a native of Zunyi, Guizhou Province. He holds a bachelor's degree and works as an engineer. His main research and professional focus is on hydroelectric operation and management.
e-mail: liujiajia19851203@outlook.com



Wanxiong MIN (born October 4, 1988), male, Han ethnicity, hails from Zunyi, Guizhou Province. He holds a bachelor's degree and is a senior engineer. His main research areas cover electrical engineering, digitalization, and algorithm models.
e-mail: wanxiong_min@outlook.com

Yuanfeng ZHONG (born January 14, 1986), male, of the Dong ethnic group, is a native of Yuping, Guizhou Province. He holds a bachelor's degree and works as an engineer. He is mainly engaged in electrical automation, automatic control, and information security.
e-mail: zyf07062023@outlook.com

An LUO (born August 17, 1987), male, Han ethnicity, is a native of Chishui, Guizhou Province. He holds a bachelor's degree and works as an engineer. He is primarily engaged in electrical engineering and its automation, electrical engineering, as well as hydropower plant operation and management.
e-mail: luoan_edu@hotmail.com