# Adaptive flow-induced vibration control using distributed sensor/actuator networks with gated recurrent units and genetic algorithms

Khalid M. Abdelaziz*[a], Jared D. Hobeck
[a]Alan Levin Dept. of Mechanical and Nuclear Engineering, Kansas State University, 1701B Platt Street, Manhattan, KS, USA 66506

## ABSTRACT

This work presents a control scheme for wind-induced vibration mitigation for tall buildings based on a gated recurrent unit (GRU) encoder-decoder model which operates using readings from multiple sensors to define a unique system state. The sensors include a distributed network of pressure probes installed on surrounding buildings, accelerometers installed on the principal building, and atmospheric conditions. The encoder-decoder GRU is trained from timeseries sensor readings to construct a unique internal representation (hidden state) of the evolving wind and building conditions. A 1:400-scale aeroelastic building model with motorized plates acting as aerodynamic control surfaces is used in wind tunnel experiments to conduct this study. An online genetic reinforcement learning (GRL) algorithm uses a series of multi-layer perceptron (MLP) networks to determine optimum actuator orientations for different flow conditions. The algorithm stores previously discovered solutions in the MLPs sorted by their fitness. The GA operates by obtaining a solution from each of the MLPs and performing GA operations on them to choose the next combination of plate angles to try. A chance also exists for trying completely random plate angles to prevent the GA from stalling. The MLPs are continuously trained during online optimization using findings obtained from new trials. The system eliminates the need for holding wind conditions, which are uncontrollable, constant during online training but still uses a pseudo-random search technique to obtain global optimum solutions. Results show a considerable reduction in building RMS acceleration when compared with a large collection of results with random constant plate angle orientations.

**Keywords:** genetic algorithms, reinforcement learning, cyber-physical systems, wind-induced vibration

## 1. INTRODUCTION

Wind-induced vibration (WIV) resistance is an important consideration in tall building design. WIV occurs due to the fluctuation in wind forces, either because of fast varying wind conditions or flow separation around bluff building geometries. WIV can seriously affect occupant comfort and render the building unserviceable or cause damage to building components such as the façade or internal walls. In severe cases, WIV can cause catastrophic failure if the structural components are not designed to withstand the resulting fatigue stresses [1]. Therefore, most design codes require buildings to withstand severe wind conditions with 50-year return periods. Compliance with these codes with respect to WIV contributes significantly to the high investment cost of tall buildings [1].

Various solutions have been developed to mitigate WIV in tall buildings. Tuned mass dampers (TMDs) consist of a large mass hanging from the top of the building like a pendulum. Tuned sloshing or liquid dampers (TLDs) consist of a liquid tank placed on top of the building that has internal flow constrictions. TMDs and TLDs operate by absorbing energy from the host structure after the vibration has started [2]. They are known to consume internal space at higher floors, where the real-estate value is usually higher. Aerodynamic modification (AM) of the building geometry is another approach of WIV reduction. It can be used independently or in conjunction with TMDs/TLDs [3]. AMs depend on changing the external geometry of the building to change the dominant wind force frequencies. The building can become less susceptible to WIV when the structural natural frequencies are located further from the dominant wind force frequencies. AMs have the capacity to inhibit even the initial formation of WIV.

Major AMs consist of changes to the overall geometry of the building such as taper, setbacks or helix shapes [4]. Minor AMs consist of chamfers, cuts or rounding at the building corners without changing the overall geometry [4]. Sharma *et al.* (2018) presented a comprehensive review of AMs applied to tall buildings. The authors concluded that minor AMs

could result in 30-60% reduction in WIV. It was also noted that minor AMs can have adverse effects depending on the local environmental conditions such as wind flow characteristics and surrounding structures [4]. This means that different wind conditions and surrounding environments require different AM configurations or orientations. Abdelaziz *et al.* (2021) performed a simulation study of a smart façade system consisting of four vertical plates installed on the corners of a building [5]. The plates could rotate around their own axis of symmetry, which provided a means for active AM. A genetic algorithm (GA) optimization procedure was used to determine the plate orientations that resulted in reduced WIV responses at different wind conditions. The optimum plate orientations were found to reduce the building deflection amplitudes by up to 80%. Abdelaziz and Hobeck (2022) performed wind tunnel experiments of a scaled building model with four vertical corner plates installed on the top section of the building [6]. An iterative optimum training (IOT) procedure was developed to optimally train an artificial neural network (ANN)-based controller using a reduced number of experiments. The controller determined the optimum plate orientations using the wind direction and time-averaged readings of the wind speed. A reduction of 40-90% in WIV acceleration was obtained when using the developed controller. The cases where synchronized wind vortices were formed saw the greatest reduction in WIV accelerations. While the developed smart façade system showed considerable effectiveness, the authors noted several concerns that may greatly hinder its adoption in real-life deployments. First, the training process was time consuming and required wind conditions to be held constant in a repeatable manner, which is not feasible for full-scale buildings. Second, potentially destructive wind conditions may need to be encountered by the training process before it can learn how to mitigate their effects. Lastly, defining wind conditions around a building with only two parameters (speed and direction) ignores many other spatiotemporal features about the wind that may be valuable for effective control. The control approach presented in this paper, addresses these concerns.

Gated recurrent units (GRUs) are a form of recurrent neural networks (RNNs) that can 'forget' irrelevant information in the hidden state and only keep information that affects the GRU's prediction capability [7]. GRUs have been utilized for wind speed forecasting, wind turbine condition monitoring, unmanned underwater vehicle state estimation, among other applications [8, 9, 10]. GRUs are less computationally expensive than long-short term memory (LSTMs), and their performance is comparable [11]. This work uses an encoder-decoder GRU model to define a unique system state using timeseries readings from multiple sensor systems. The sensors include a distributed network of pressure probes installed on surrounding structures, accelerometers installed on the principal building, and an atmospheric conditions sensor. Overall, the GRU acts as a time-sensitive encoder for the wind-building system. The use of the sensor systems in conjunction with the GRU model eliminates the need to directly measure the wind speed and direction while providing a prediction window in which the system can act to prevent incoming hazardous wind events. Finally, the inclusion of the temporal depth of the sensor readings exposes the GRU model to various patterns of system behavior that can be common among different wind conditions (e.g., vortex structures). This is expected to improve the model's ability to generalize to wind conditions it did not specifically encounter during training.

One of the challenges to the original smart façade system was the requirement to hold the wind conditions constant until the system learns how to mitigate the WIV effects, i.e., forcing the environment to be static. Alternatively, reinforcement learning (RL) offers a way of programming agents through trial-and-error in a dynamic environment. For each state the agent encounters, RL training presents the agent with a set of possible actions. After the agent performs one of the actions, the RL training rewards it in proportion to how favorable the resulting system state was. Deep RL algorithms attracted increasing interest from machine learning researchers. It was successfully used for autonomous driving, learning to walk, human-level performance in 3d multiplayer games, path planning for unmanned aerial vehicles, among others [12, 13, 14, 15]. Despite its success, efficient exploration of complex domains remains challenging for RL algorithms [16, 17]. This limitation can make the RL-generated policy unable to provide global optimum solutions especially for systems with infinite possible states and actions, such as the wind-building-plates system at hand. It is known that greater WIV acceleration response is coincident with the formation of vortices that eventually synchronize with the building motion (i.e., aeroelastic *lock-in*) [4, 6]. Therefore, it is desirable to generate an agent able to obtain global minimum WIV accelerations in one plate rotation action, rather than a series of actions, to minimize the possibility of aeroelastic lock-in.

This work presents a genetic reinforcement learning (GRL) algorithm which enhances RL exploration and enables it to more quickly obtain global optimum solutions. GRL employs genetic algorithms (GAs) integrated with a series of trainable multi-layer perceptron (MLP) policy functions. The MLP policy functions take the current GRU hidden state as an input and provide the plate orientations dictated by the policy as an output. The genetic reinforcement learning (GRL) procedure starts by sorting the current solution (plate orientations) into the set of MLP policies according to how favorable the recorded building accelerations were. The first MLP policy is always trained using the best-found solutions for the encountered GRU hidden states, the second MLP policy is trained using the second-best solutions, and so on. To determine

the plate orientations to explore next, the solutions suggested by all MLP policies for the current hidden state are combined to form a population of solutions to perform GA operations on. While GAs inherently provide means for controlling exploration and exploitation, a chance for trying fully random solutions is also added to further enhance this control. As the training progresses, each MLP policy represents a different 'front' of solution quality which provides a mechanism for elitism while maintaining solution diversity. GRL can still find better solutions for states it did not exactly encounter because both the GRU and the MLP policies have inherent interpolation ability. During operation, only the best MLP policy (trained from best-found solutions) is used to determine the plate orientations to use for the current GRU hidden state. The developed procedure eliminates the need to keep the uncontrollable wind conditions constant while performing a pseudo-random search capable of obtaining minimum WIV accelerations in one plate rotation action.

This paper is organized as follows: Section 2 details the experimental setup and the problem definition. Section 3 explains the methods used to train the encoder-decoder and the GRL procedure. Section 4 illustrates the obtained WIV mitigation results and compares to the established baselines. Finally, Section 5 lists the conclusions of this work and possible future extensions and improvements.

## 2. EXPERIMENTAL SETUP

### 2.1 Overview

The Commonwealth Aeronautical Advisory Research Committee (CAARC) standard building model is used in this study because it is thoroughly studied in the literature. The building has the dimensions 180 × 45 × 30 m (Height H× Width B× Depth D) with an average density of 160 kg/m$^3$. Figure 1 illustrates the wind tunnel experimental setup which uses a 1:400 scaled robotic aeroelastic model (450 × 112.5 × 75 mm) that was designed and manufactured by Abdelaziz and Hobeck [6]. The model has 4 lumped masses (floor plates + added masses) that distribute the weight equally along the height. The floors are connected only by 4 columns (threaded rods) that have variable diameters along the height. The top floor accounts for two consecutive floors in the mass budget because it includes the motors, the rotating plates and their accessories. The model was designed using a genetic algorithm (GA) optimization that matched the dynamic characteristics of the scaled model with the those of the full-scale building [6]. Figure 1 also shows the degrees of freedom (DOFs) of interest: along-wind sway (bending, $x$), cross-wind sway (bending, $y$) and torsional (twisting, $\theta$) along with the wind speed direction $U$. This study considers only this direction of wind speed because it was found to be the most severe in previous research studies [6]. The scaled building model has four corner plates with orientation angles $\alpha_1 \dots \alpha_4$ as they can freely rotate around their axis of symmetry. The length of the plates covers about 30% of the building height to minimize the investment costs associated with employing the system. Separation of the air flow happens at the corners and wind forces cause the greatest bending and twisting moments at the top, so the plates were placed at these locations to maximize their effect on the wind-building system. The experimental setup includes a rigid building model with dimensions 400 × 85 × 75 mm placed upstream of the aeroelastic model. This rigid building is included to test the system's ability to handle complex wind flow patterns, such as turbulent wakes and vortex structures.

### 2.2 Sensor network

Figure 1 also shows a pressure sensor network consisting of 5 differential pressure sensors that were fitted inside 3d-printed housings with extension tubes on the high-pressure port of the sensors so they can sense the wind flow. The low-pressure port is contained inside the housing to sense the static pressure value. All dimensions are measured relative to the tips of the sensors indicated in Figure 1. The wind tunnel floor is used as a reference for all height measurements. Sensor 1 is installed inside the upstream building at a height of 380 mm. Sensor 2 is installed on top of another rigid foam cube at a height of 75 mm. Sensors 3 and 4 are oriented at 45° from the wind direction $U$ and installed on a steel tube extending from the tunnel floor at a height of 200 mm. Sensor 5 is installed on another similar tube at a height of 360 mm. The placement and count of the sensors have a strong effect on the encoder-decoder GRU's ability to fully define the system state. For this initial study, the sensors were placed using human experience. For example, sensor 3 is positioned to endure most of the vortex shedding effect from the upstream rigid building. Though the current placement is shown later to provide acceptable predictions from the encoder-decoder GRU, the optimal placement of the sensors is still being considered for future studies. In addition to the pressure sensors, two accelerometers are installed on the top floor of the scaled building model [6]. Their readings are combined to obtain the accelerations for the 3 DOFs of interest. Finally, an ambient atmospheric conditions sensor is installed outside the wind tunnel to measure temperature, humidity and barometric pressure. As discussed in Section 3.1, the collection of all sensor readings is used in conjunction with the encoder-decoder GRU to define the system state.

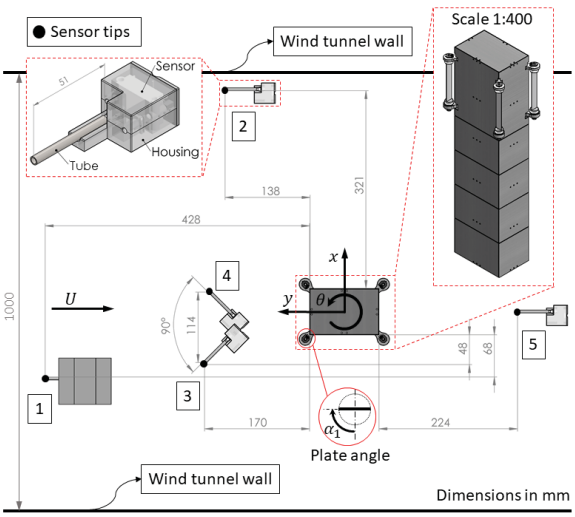


Figure 1. A top view of the wind tunnel experiment setup showing: (1) the 1:400 scaled aeroelastic model centered in the width of the tunnel with four rotating plates installed at its corners; (2) the measurement conventions for the DOFs of interest ($x$, $y$ and $\theta$), plate angles ($\alpha_1 \dots \alpha_4$) and the wind direction $U$; and (3) the upstream building and the pressure sensor locations and shape.

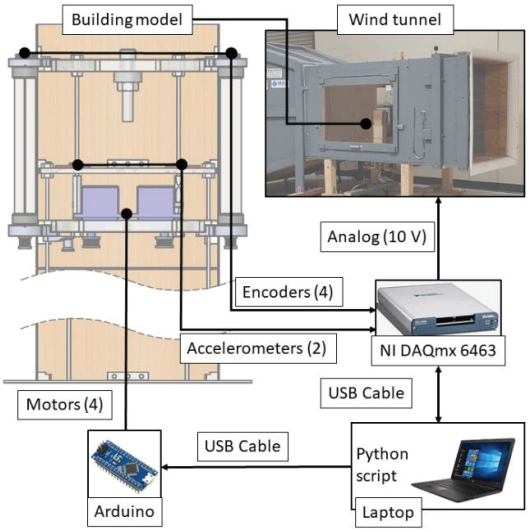


Figure 2. A schematic showing the various components of the cyber-physical setup for the wind tunnel experiments. The python script controls all aspects of the online training/control experiment.

### 2.3 Cyber-physical setup

Figure 2 illustrates the cyber-physical setup used to implement the training and control procedures described in Section 3. In addition to sensors described in the previous section, four rotational position encoders are installed on the free end of the plates to provide their current position. The plate position readings are not directly used by the encoder-decoder GRU to determine the system state to avoid redundancy with the building accelerations (which partly result from the plate positions). The plate position readings are used by the controller as described in Section 3.3 to determine the current plate angles. All sensors and encoders provide analog outputs which are connected to an NI DAQmx® 6463 data acquisition unit. The sampling rate is set at 200 Hz which is determined from the maximum response time of the Sensirion SPD810 differential pressure sensors which were used. The plate motors are controlled by an Arduino® microcontroller. A python script controls the data acquisition through the nidaqmx software library, the wind tunnel speed through analog output and the plate motors through a serial connection to the Arduino microcontroller. The script also implements the online training and control procedures described in the next section. This includes tasks such as timing the experiments, data collection, experiment numbering and online training.

## 3. CONTROLLER DESIGN

### 3.1 Encoder-decoder GRU

Figure 2 shows the encoder-decoder GRU model. The inputs to the model are the sensor readings $\delta_0 \dots \delta_n$ and the outputs are the future sensor readings $\delta_{n+1} \dots \delta_{n+m}$ for $n$ input and $m$ output time steps, respectively. The model uses the inputs to populate an internal hidden state array $S$. The sensor readings at time $t$, $\delta_t$ are:

$$\delta_t = [p_0 \quad p_1 \quad p_2 \quad p_3 \quad p_4 \quad p_b \quad T \quad H \quad \ddot{x} \quad \ddot{y} \quad \ddot{\theta}]_t \tag{1}$$

where $p_0 \dots p_4$ are the pressure sensor readings (see Figure 1), $p_b$ is the barometric pressure, $T$ is the ambient temperature, $H$ is the humidity, $\ddot{x}$, $\ddot{y}$ and $\ddot{\theta}$ are the acceleration components indicated in Figure 1. The encoder portion of the model is stateful, so the hidden state $S$ accumulates its information from all previous inputs as well as the current input. The architecture for the encoder-decoder GRU follows a standard powers-of-2 units for each layer.

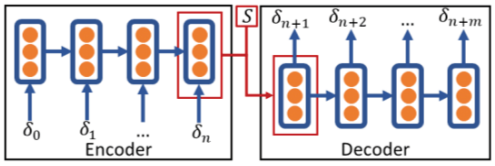


Figure 3. A schematic diagram of the encoder-decoder GRU model indicating the sensors inputs $\delta_0 \dots \delta_n$, outputs $\delta_{n+1} \dots \delta_{n+m}$ and the hidden state $S$.

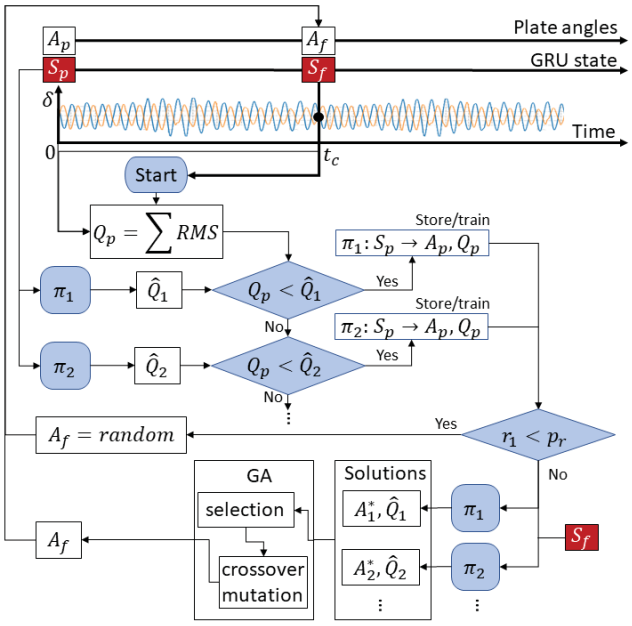


Figure 4. The modified RL procedure indicating how the previous plate angles $A_p$, GRU state $S_p$ and acceleration sum $Q_p$ (Equation 2) are used to train the policy MLPs ($\pi_k$). The policy MLPs are then used to determine the future plate angles $A_f$ using the future GRU state $S_f$.

The encoder has 4 stateful GRU layers with 16, 32, 64 and 256 units respectively. The decoder has the same number of layers, but the unit count is reversed. The number of layers was obtained by trial until an acceptable prediction performance was reached. The encoder-decoder GRU is trained from randomly collected wind tunnel data. The prediction (decoder) portion is not used after initial training. It is only used so the training process can automatically prune the hidden state $S$ to only keep information that is relevant for fully defining the system state.

### 3.2 Controller objective

The desired function of the controller is to obtain plate orientations that minimize WIV accelerations at different wind conditions, i.e.:

$$Min.\, Q(S,A) = RMS(\ddot{x}) + RMS(\ddot{y}) + \frac{B}{2} RMS(\ddot{\theta}) \qquad (2)$$

In reference to Figure 1 and Equation 1, $A = [\alpha_1 \quad \dots \quad \alpha_4]$ are the plate angles (0-180°). $S$ is the current system state. RMS is the root-mean-square, and $\ddot{x}$, $\ddot{y}$ and $\ddot{\theta}$ are the accelerations of the DOFs of interest. The RMS of $\ddot{\theta}$ is numerically scaled by half of the building width $B$ (the radius of twist) to have the same units as the other components. The range of 0° to 180° for $\alpha_i$ covers all the possible solutions because the plates are aerodynamically symmetric. If nonsymmetric plates (e.g., wing profiles) are used, then a 360° range should be considered.

### 3.3 Genetic reinforcement learning algorithm

Figure 4 illustrates the GRL algorithm that runs after each control interval $[0, t_c]$. At the point of control (indicated by an arrow to the 'Start' bubble), the previous plate angles $A_p$ and GRU state $S_p$ are known. Also, the future GRU state $S_f$ is obtained by feeding the previous sensor information $\delta_0 \dots \delta_{t_c}$ to the GRU. The algorithm is required to do the following: (1) use the previous information to train the MLP policies ($\pi_1, \pi_2, \dots$) and (2) chose which future plate angles $A_f$ to try. The policies have the form $\pi(S) = [A, Q]$, so they not only provide the plate angles dictated by the policy, but also the expected RMS sum when these angles are used. The procedure starts by calculating Equation 2 ($Q_p$) from previous sensor information (accelerations only in Equation 1). Starting from $\pi_1$, the procedure compares the calculated RMS sum $Q_p$ with

the policy's prediction $\hat{Q}_1$. If the calculated sum is smaller, the training data is added to the storage for the current policy only and the comparison is not performed for the other policies. The accumulated effect of these comparisons is that the best solutions found would be stored for $\pi_1$, and solutions that perform worse would be stored for $\pi_2$, and so on. Each of the policies $\pi$ is trained frequently using the collection of data stored for it.

To determine the next plate angles to try $A_f$, a random number is generated and compared with the probability $p_r$. If the random number is smaller than $p_r$, $A_f$ is set to a completely random value to enhance exploration. Otherwise, each of the policies provides optimum angles $A_k^*$ and expected RMS sum $\hat{Q}_k$ for the future GRU state $S_f$. A single GA iteration is performed on the generated solutions with crossover and mutation probabilities $p_c$ and $p_m$, respectively. The future plate angles $A_f$ are chosen at random from the GA result. The GA selection (tournament selection) makes it more probable that better solutions would be chosen, so the accumulated effect of the GA iteration is to provide a tendency to try solutions expected to provide better performance for the future state $S_f$. The probabilities $p_r$, $p_c$ and $p_m$ provide convenient control of the balance between exploitation and exploration. During operation (after training), $\pi_1(S_f)$ is evaluated directly and $A_f$ is set to $A_1^*$.

# 4. RESULTS AND DISCUSSION

## 4.1 Encoder-decoder GRU prediction

The time step for the encoder-decoder GRU model was set to 0.01 s and the sensor readings were down-sampled to match it. The training data is collected from constantly varying random wind conditions and plate orientations. Therefore, increasing the input/output time steps (prediction time) negatively impacts the prediction performance when the condition variations no longer have precursors in the input data. For this study, the prediction time was increased gradually starting from 1 second until the prediction performance started to diminish. The prediction time reached 7.5 s (750 time steps) without considerable impact to the prediction performance. The time scale with the full-scale building is 1/33, so 7.5 seconds equals approximately 4 minutes for the full-scale building. This gives the control system enough time to respond to most wind events [6]. Figure 5 compares the predicted and the actual rotational acceleration $\ddot{\theta}$ for a 16-minute data set collected from the wind tunnel. The figure also zooms on a full 7.5 seconds of encoder-decoder output. For the full 16 minutes of prediction, the mean absolute error was only 3.5% of the maximum rotational acceleration. The rotational acceleration $\ddot{\theta}$ is used in this comparison because it was found to be the most significant acceleration component. The comparison confirms the encoder-decoder model's ability to predict and thus fully define the system in its hidden state.

## 4.2 Controller performance

In reference to Figure 4, Table 1 lists the parameters used to test the GRL algorithm. The control interval $t_c$ was set by observing the stability of the RMS accelerations at constant wind and plate conditions. When the RMS acceleration becomes constant, it means that the important oscillatory features have already been captured in the considered control length.

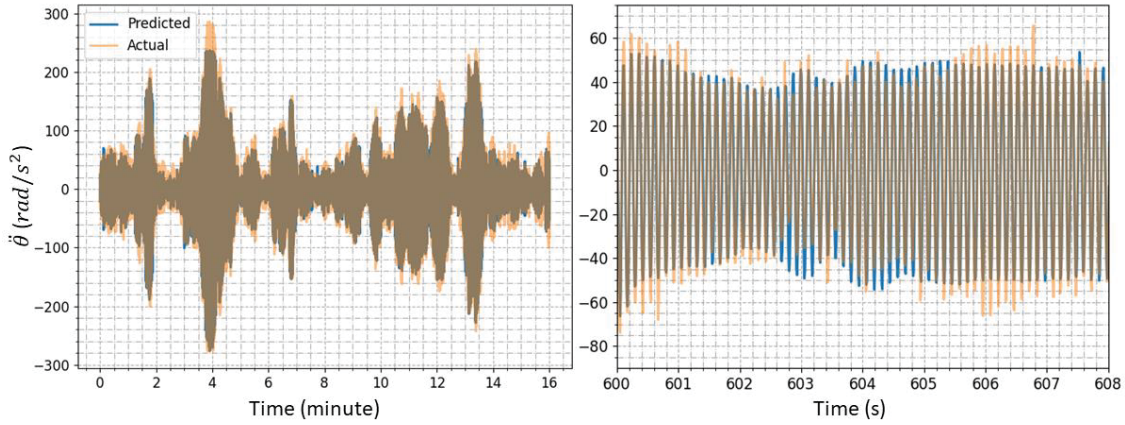


Figure 5. A comparison between the predicted and actual building rotational acceleration $\ddot{\theta}$ for 16 minutes and 7.5 seconds of data.

Table 1. GRL ALGORITHM PARAMETERS

| Parameter | Value |
|---|---|
| **Control interval $t_c$** | 15 s |
| **Policy MLP count $k$** | 7 |
| **Random probability $p_r$** | 0.3 |
| **Crossover probability $p_c$** | 0.8 |
| **Mutation probability $p_m$** | 0.2 |

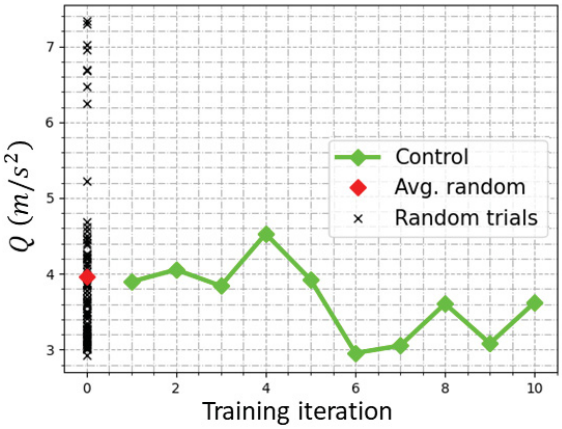


Figure 6. Evolution of the controller WIV mitigation performance during 10 iterations of the GRL algorithm subjected to randomly generated wind speed profiles compared to baseline data created with 90 combinations of static (not controlled) plate angles and random wind speed profiles.

The control interval is double the GRU input/output time, so the GRU is dispatched two times per control interval to construct the hidden state. The MLP policy count $k$ was initially set to 12, but the GRL algorithm only managed to store data for 7 out of 12 policies. The process was then repeated using only 7 policy MLPs. The random probability $p_r$ was set at 0.3 for this initial study. In future studies, $p_r$ may be set to 1.0 until the policies have enough data points to construct reasonable predictions. The crossover and mutation probabilities $p_c$ and $p_m$ are typical values for GAs.

The procedure illustrated in Figure 4 was run on random variations of the wind tunnel speed. The intervals of the speed changes as well as the speed values were randomized. The length of every generated wind speed profile was 10 minutes. Figure 6 illustrates the change in the RMS sum $Q$ (Equation 2) across subsequent GRL training iterations which are run against different wind speed profiles. To establish a baseline for the controller performance, random plate angle combinations were tested against the same wind speed profiles. For each 10-minute randomly varying wind profile, the plate angles were positioned at a constant (not controlled) random position. This was done for 90 different combinations of wind profiles and plate angles. The RMS sum $Q$ values for each random case is plotted in Figure 6. It is important to note that the minimum $Q$ value attained from these static tests corresponds to a specific combination of plate angles and random wind profile. If a different wind profile was used for this same set of plate angles, the $Q$ value will certainly increase, thus motivating the need for an active (rather than static/passive) façade system. Figure 6 also shows that the controller performance indicated by the green training points improves (decreases) across iterations despite being subjected to all random wind speed profiles. The RMS sum $Q$ points for the controller during training does not subject the building to worst-case plate angles, which is an important practical consideration and large improvement from previous controller designs. The controller manages to surpass the performance of the best random plate angles after 5 training iterations. These findings confirm the effectiveness of the GRL algorithm. After the controller was trained for 10 iterations, several aspects of its performance are discussed herein.

Figure 7 focuses on two different timeseries locations of interest. In both cases, the wind speed $U$ is changing in a way that causes the acceleration response to increase. This is also evident from the RMS acceleration plot. As soon as the acceleration's control interval $t_c$ passes, it detects this increase and repositions the plates according to the trained policy MLP $\pi_1$. As a result, the acceleration response decreases even though the wind speed keeps increasing. This experiment confirms that the controller can anticipate incoming wind events and adjust for them before accelerations become severe. This prevents phenomena such as aeroelastic lock-in from occurring.

To further illustrate the WIV mitigation mechanism, Figure 8 compares the RMS rotational acceleration $\ddot{\theta}$ with and without using the trained controller. The RMS is calculated using a 20-second window. The wind speed profile is randomly generated but constant for both cases to make a fair comparison. The comparison shows that the controller reduces RMS rotational acceleration by up to 75% and maintains accelerations significantly lower than the uncontrolled case for a vast majority of the time.

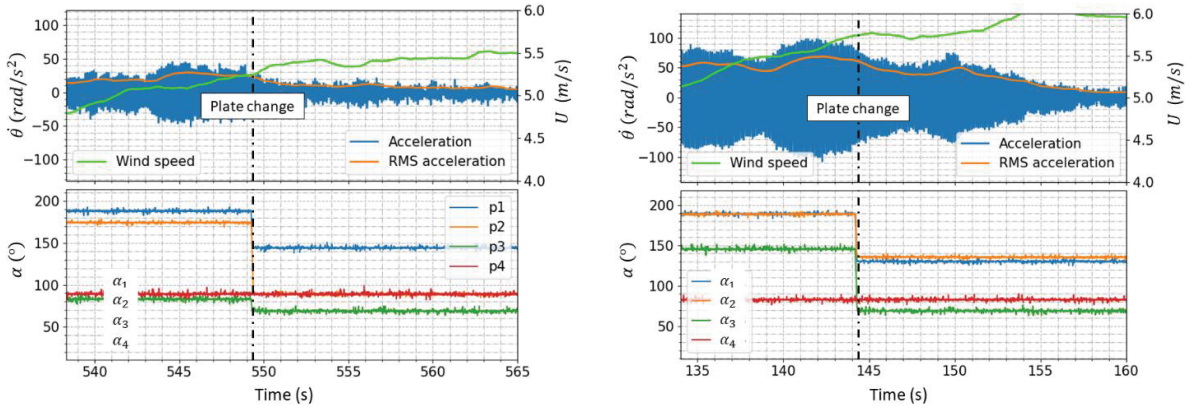


Figure 7. Two different cases where the trained controller automatically responded to the increase in the rotational acceleration $\ddot{\theta}$ (resulting from the change in wind speed $U$) by changing the plate angles which immediately reduced the rotational acceleration.

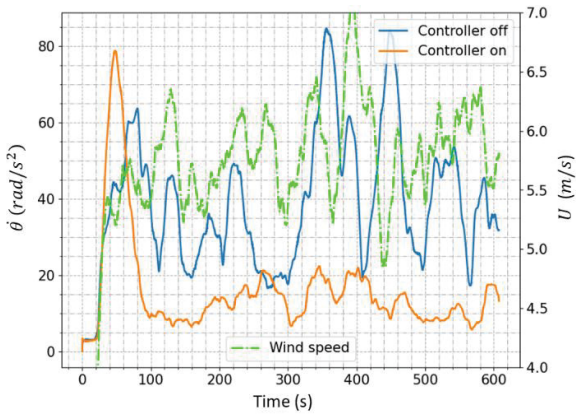


Figure 8. A comparison of the RMS rotational acceleration (20-second window) for the same randomly varying wind speed profile when the controller is turned off and on.

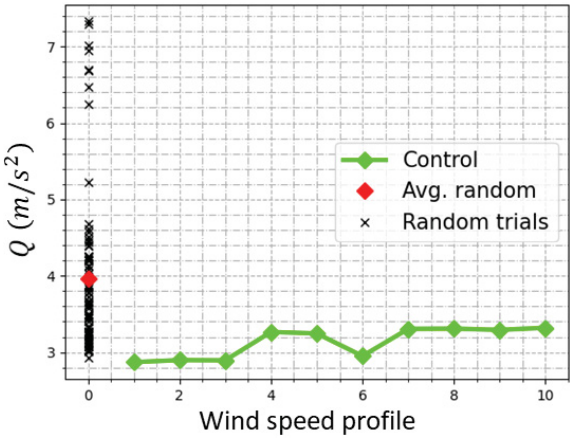


Figure 9. The effect of changing the wind speed profile on the controller performance $Q$ compared with the performance of the evaluated random plate angles.

To get an overall representation of the trained controller performance across many conditions, Figure 9 tests the trained controller by subjecting it to multiple wind speed profiles and compares the RMS sum performance with the random baseline. This differs from Figure 6, which focuses on the WIV acceleration response during training. It is shown that the controller performance is not strongly affected by changing the wind conditions. This proves that the controller trained using the GRL algorithm can generalize to conditions it did not specifically encounter during training. This is an important practical consideration because the real-life conditions are infinite and training for all of these conditions is not feasible.

## 5. CONCLUSIONS

This work presents a control system for wind-induced vibration mitigation in tall buildings. The system uses four plates that are able to rotate about their axes of symmetry to function as aerodynamic control surfaces. A technique that enables defining fast varying and complex wind conditions is presented. A sensor network consisting of a set of pressure probes, accelerometers and ambient conditions sensor are used as time series inputs to an encoder-decoder GRU model. The model uses the inputs to define an instantaneous hidden state that fully defines the wind-building system state and 'remembers' long-term effects relevant for predicting future sensor readings. A genetic reinforcement learning procedure is developed to use the encoder-decoder hidden state to perform pseudo-random optimization in a highly dynamic environment. The

process used a series of trainable policy MLP functions as ‘agents’ that can provide a genetic algorithm with a set of diverse solutions. The control system was tested on a 1:400 scale robotic aeroelastic building model in wind tunnel experiments. The important findings of this work are summarized below:

1. The encoder-decoder GRU model was trained from wind tunnel experiment data and was able to predict the future rotational acceleration from the current rotational acceleration with an average absolute error that is only 3.5% of the maximum encountered acceleration. This confirms that the encoder-decoder GRU hidden state fully defines the wind-building system.
2. The GRL algorithm managed to considerably improve the controller performance within only five 10-minute training iterations. During training, the algorithm did not subject the building to severe catastrophic conditions such as encountered by some of the baseline random plate angles.
3. The trained controller reduced the RMS rotational acceleration by up to 75% for the same test wind speed profile.
4. The trained controller was able to detect adverse wind conditions before the peak wind speed occurred. So, it can anticipate severe conditions and adjust for them before they happen.
5. The controller performance was not affected considerably when tested against a group of random wind speed profiles, which confirms its ability to generalize to conditions it was not specifically trained for.

In future work, a sensitivity analysis of the different controller parameters will be carried out to establish guidelines for setting these parameters. This includes the number of iterations to train the controller for, among others. Comparison with other reinforcement learning techniques and variations in the literature is also an area of potential future work.

## REFERENCES


[1] J. S. Love, T. C. Haskett and B. Morava, "Effectiveness of dynamic vibration absorbers implemented in tall buildings," *Engineering Structures,* vol. 176, pp. 776-784, 2018.

[2] A. Y. Tuan and G. Q. Shang, "Vibration Control in a 101-Storey Building Using a Tuned Mass Damper," *Journal of Applied Science and Engineering,* vol. 17, no. 2, p. 141–156, 6 2014.

[3] K. T. Tse, P. A. Hitchcock, K. C. S. Kwok, S. Thepmongkorn and C. M. Chan, "Economic perspectives of aerodynamic treatments of square tall buildings," *Journal of Wind Engineering and Industrial Aerodynamics,* vol. 97, pp. 455-467, 2009.

[4] A. Sharma, H. Mittal and A. Gairola, "Mitigation of wind load on tall buildings through aerodynamic modifications: Review," *Journal of Building Engineering,* vol. 18, pp. 180-194, 2018.

[5] K. M. Abdelaziz, A. Alipour and J. D. Hobeck, "A smart façade system controller for optimized wind-induced vibration mitigation in tall buildings," *Journal of Wind Engineering and Industrial Aerodynamics,* vol. 212, p. 104601, 2021.

[6] K. M. Abdelaziz and J. D. Hobeck, "Optimum aeroelastic control via iterative neural network training for wind-resistant cyber–physical buildings," *Applied Soft Computing,* vol. 114, p. 108100, 2022.

[7] K. Cho, B. van Merrienboer, D. Bahdanau and Y. Bengio, "On the Properties of Neural Machine Translation: Encoder-Decoder Approaches," *CoRR,* vol. abs/1409.1259, 2014.

[8] D. Wei, J. Wang, X. Niu and Z. Li, "Wind speed forecasting system based on gated recurrent units and convolutional spiking neural networks," *Applied Energy,* vol. 292, p. 116842, 2021.

[9] Z. Kong, B. Tang, L. Deng, W. Liu and Y. Han, "Condition monitoring of wind turbines based on spatio-temporal fusion of SCADA data by convolutional neural networks and gated recurrent units," *Renewable Energy,* vol. 146, pp. 760-768, 2020.

[10] C. Lin, H. Wang, M. Fu, J. Yuan and J. Gu, "A Gated Recurrent Unit-Based Particle Filter for Unmanned Underwater Vehicle State Estimation," *IEEE Transactions on Instrumentation and Measurement,* vol. 70, pp. 1-12, 2021.

[11] A. Rehmer and A. Kroll, "On Using Gated Recurrent Units for Nonlinear System Identification," in *2019 18th European Control Conference (ECC)*, 2019.

[12] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. A. Sallab, S. Yogamani and P. Pérez, "Deep Reinforcement Learning for Autonomous Driving: A Survey," *IEEE Transactions on Intelligent Transportation Systems,* pp. 1-18, 2021.

[13] T. Haarnoja, S. Ha, A. Zhou, J. Tan, G. Tucker and S. Levine, *Learning to Walk via Deep Reinforcement Learning,* 2019.

[14] M. Jaderberg, W. M. Czarnecki, I. Dunning, L. Marris, G. Lever, A. G. Castañeda, C. Beattie, N. C. Rabinowitz, A. S. Morcos, A. Ruderman, N. Sonnerat, T. Green, L. Deason, J. Z. Leibo, D. Silver, D. Hassabis, K. Kavukcuoglu and T. Graepel, "Human-level performance in 3D multiplayer games with population-based reinforcement learning," *Science,* vol. 364, pp. 859-865, 2019.

[15] C. Qu, W. Gai, M. Zhong and J. Zhang, "A novel reinforcement learning based grey wolf optimizer algorithm for unmanned aerial vehicles (UAVs) path planning," *Applied Soft Computing,* vol. 89, p. 106099, 2020.

[16] Z.-W. Hong, T.-Y. Shann, S.-Y. Su, Y.-H. Chang and C.-Y. Lee, *Diversity-Driven Exploration Strategy for Deep Reinforcement Learning,* 2018.

[17] E. Conti, V. Madhavan, F. P. Such, J. Lehman, K. O. Stanley and J. Clune, *Improving Exploration in Evolution Strategies for Deep Reinforcement Learning via a Population of Novelty-Seeking Agents,* 2018.