Editors' Suggestion

Machine-learning-assisted tracking of magnetic objects using quantum diamond magnetometry

Fernando Meneses,[1,2,*] Christopher T.-K. Lew,[1,3,*] Anand Sivamalai,[4] Andy Sayers,[4] Brant C. Gibson,[3] Andrew D. Greentree,[3] Lloyd C. L. Hollenberg,[1,2] and David A. Simpson[1,5,†]

[1]*School of Physics, University of Melbourne, Parkville, Victoria 3010, Australia*
[2]*Centre for Quantum Computation and Communication Technology, School of Physics, University of Melbourne, Parkville, Victoria 3010, Australia*
[3]*School of Science, RMIT University, Victoria, 3001, Australia*
[4]*Phasor Innovation Pty Ltd, 155 Straws Lane, Hesket, Victoria 3442, Australia*
[5]*ARC Centre of Excellence in Quantum Biotechnology, School of Physics, University of Melbourne, Parkville, Victoria 3010, Australia*

(Received 8 December 2025; revised 17 June 2026; accepted 27 August 2026; published 2 November 2026)

Remote magnetic sensing can be used to monitor the position of objects in real time, enabling ground transport monitoring, underground infrastructure mapping, and hazardous detection. However, magnetic signals are typically weak and complex, requiring sophisticated physical models to analyze them and a detailed knowledge of the system under study, factors that are frequently unavailable. In this work, we provide a solution to these limitations by demonstrating a machine-learning (ML) method that can be trained exclusively on experimental data, without the need of any physical model, to predict the position of a magnetic target in real time. The target can be any object with a magnetic signal above the floor noise, and in this case, we use a quantum diamond magnetometer to track variations of a few hundred nanoteslas produced by an elevator moving along a single axis. The one-dimensional movement is a simple yet challenging scenario, resembling realistic environments such as high buildings, tunnels, or train circuits, and is the first step toward building broader applications. Our ML algorithm can be trained in approximately 40 min, achieving over 80% accuracy in predicting the target's position at a rate of 10 Hz, for a positional error tolerance of 30 cm, which is well within the the 4-m spacing between parking levels. Our results open up the possibility to apply this ML method more generally for real-time monitoring of magnetic objects, which will broaden the scope of magnetic detection applications.

DOI: 10.1103/vjtb-d8kb

I. INTRODUCTION

Real-time object monitoring plays an essential role across many fields, such as traffic monitoring [1,2], environmental control [3], and surveillance [4–6]. Most well-established techniques are based on the global positioning system (GPS), which relies on radio-wave signals and satellite triangulation [7]. Although a powerful technique, GPS has important limitations: first, it requires that the object under tracking be equipped with a GPS receiver, thus restricting use to cooperative scenarios. Second, there are several GPS-denied environments where radio-wave signals are blocked or distorted, such as underwater, underground, and indoor spaces [8,9]. For all these cases, a different technology is needed, and this is where remote magnetic sensing comes into play, as magnetic signals can propagate in GPS-denied environments and are generated directly by the tracked object, thus not limited to cooperative scenarios.

The major challenge for remote magnetic sensing is interpreting the data, as each tracked object will have its own magnetic signature, convoluted with the environmental noise. The conventional approach for analyzing this kind of information is to build a physical model that describes the target and its environment, and then fit the experimental data to extract the position information. Examples of this method can be found in submarine navigation [10,11], detection and classification of ground vehicles [12], detection of magnetic anomalies [13], and moving magnetic targets [14]. Recent works have introduced ML tools to complement the data analysis and improve the performance compared to classical

*These authors contributed equally to this work.
†Contact author: simd@unimelb.edu.au

Published by the American Physical Society under the terms of the Creative Commons Attribution 4.0 International license. Further distribution of this work must maintain attribution to the author(s) and the published article's title, journal citation, and DOI.

algorithms, but they still rely on physical models to preprocess magnetic data or synthesize training datasets. For instance, Wu *et al.* developed a deep-learning model to monitor the activity of underground trains [15]; Cárdenas *et al.* used convolutional neural networks (CNNs) to detect magnetic anomalies and identify the localization, number, and properties of magnetic dipoles [16]; Sun *et al.* employed deep neural networks (DNNs) to localize complex arrays of underground pipelines [17]; Wang *et al.* used DNNs integrated with a moving magnetometer to track magnetic anomalies [18]; Vandavasi *et al.* improved underwater navigation with shallow learning AI algorithms [19]; and Meng *et al.* have used the objective base function optimization method to develop a machine-learning algorithm integrated with a vector magnetometer [20].

Although magnetic data interpretation has made remarkable progress, most of the real-world scenarios display too complex magnetic signals, which would require extensive physical models. As the models grow in complexity, more prior information about the magnetic system is needed in order to determine numerous parameters and fit the experimental data. As a result, remote magnetic sensing becomes unfeasible in most situations where limited information is available. As such, much research has focused on the development of modeling-free alternatives for interpreting the magnetic data, powered by artificial intelligence. The main idea is that experimental measurements can be analyzed by ML algorithms that work as "black boxes," meaning that they make their own (hidden) model for interpreting data, ultimately taking magnetic measurements as inputs and returning the position of the tracked object. Some examples of these experimentally trained ML algorithms include the recognition of submarine cables [21]; robot's indoor magnetic navigation [22]; and identification of basic maneuvers of ground vehicles [23]. Research in this area has focused on on object recognition, autonomous navigation, or qualitative behavior, but none of them have the ability to monitor the location of a moving object in real time.

In this work, we develop an experimentally trained ML algorithm that predicts the position of an elevator in an indoor environment in real time, using magnetic measurements as the input data, without the need of developing any physical model to interpret the magnetic information. The target (i.e., the elevator car) can move along a vertical axis across eight levels spaced 4-m apart at an average speed of 1.4 m/s, producing variations of a few hundreds of nanoteslas in the measured magnetic signals. Although this one-dimensional movement may seem simple, the magnetic system is in fact complex and involves multiple elements, such as the elevator car, counterweight, steel cables, and electrical motor, each one contributing with its own magnetic signature. In addition, these signals are convoluted with the strong magnetic fields coming from nearby electrical trams, which we identify as magnetic noise for our tracking purposes. As explained above, building a physical magnetic model for this real-world problem is a challenging task that requires a significant level of prior knowledge about the elevator itself and its environment.

In situations where prior knowledge is limited, we need a highly sensitive magnetic field sensor that can operate in real-world scenarios. In this work, we utilize a quantum diamond magnetometer based on a dense ensemble of nitrogen-vacancy (NV) defects as our sensing platform [24–29]. Our magnetometer can operate in a wide temperature range without any magnetic shielding, detecting full-vector magnetic fields with an average sensitivity of around 400 $\mathrm{pT}/\sqrt{\mathrm{Hz}}$ per Cartesian axis within a 1–100 Hz measurement bandwidth. Furthermore, quantum diamond magnetometers have inherent capabilities promising accurate and stable measurements during long periods of time without any drift or the need for recalibration.

Our results demonstrate that with 6 h of training with experimental results, the ML algorithm can predict the position of the elevator car, represented by its lowest point, at a 10 Hz rate with an accuracy of 80% for a positional error tolerance of 30 cm. As this distance is much smaller than the 4-m spacing between parking levels, and the accuracy implies that on average 8 out of 10 points are correct every second, the trajectory of the elevator and the arrival at each parking level can be reliably monitored in real time. The flexibility of both our ML algorithm and magnetic platform allows for easy adaptation to various scenarios and monitoring tasks, promising broad applicability in situations where only experimental data are available.

II. RESULTS

A. Application overview

The principle of the real-time object monitoring using the NV quantum diamond magnetometer is illustrated in Fig. 1. Our target is an elevator car, with XYZ dimensions $1.8 \times 2.0 \times 2.6$ m^3, that can move along eight levels on the Z axis, with levels spaced 3.7 m apart except for the lowest two, spaced by 4.1 m. The elevator travels at an average speed of 1.4 m/s, meaning that it typically takes almost 3 s to move between adjacent levels. The quantum diamond magnetometer is located at the basement (lowest level), approximately 10 m from the elevator, and it detects variations of the magnetic field $\vec{B}$ when the elevator is moving, which are in the order of a few hundreds of nT. However, the magnetic signal does not exclusively reflect the magnetic field from the car alone, but also the Earth's magnetic field (approximately 50 μT) and contributions from other magnetic elements in the elevator system, mainly the counterweight (traveling in the opposite direction to the car), the steel cables and the electrical motor located on top of level 8 [see Fig. 1(a)]. Additionally, there

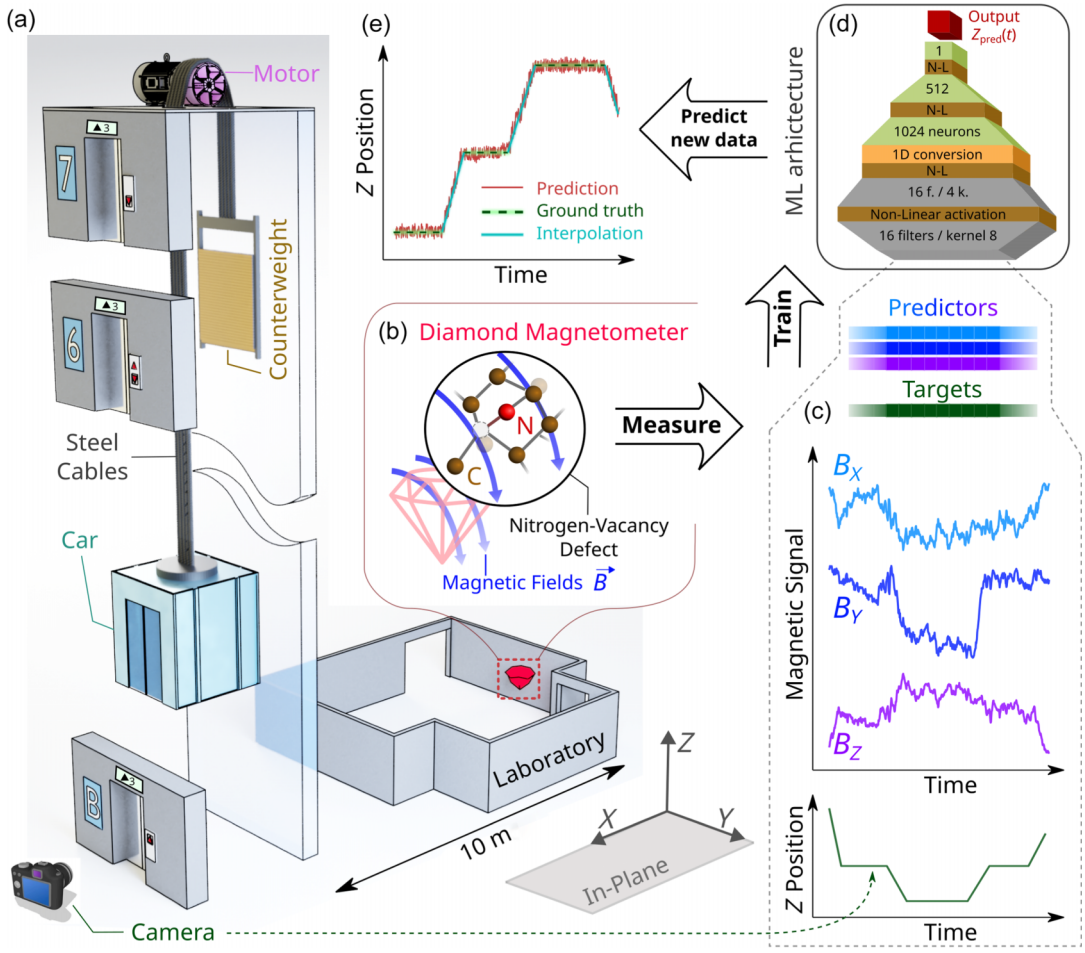


FIG. 1. Real-time object monitoring: application overview. (a) Scheme of the elevator system, where the main magnetic elements are identified: target car, counterweight, steel cables, and electrical motor. The sensing platform is located in the lowest level of the building, 10 m away from the elevator's doors. The position of the car is recorded independently by a camera placed in front of the informative display. (b) Quantum diamond magnetometer based on a dense ensemble of nitrogen-vacancy defects, capable of sensing external magnetic fields. The inset shows the diamond crystallographic structure containing the NV defect. (c) Set of magnetic measurements and Z positions of the elevator's car as a function of time. They are processed into a one-dimensional timeseries as predictors and targets, respectively. (d) Machine-learning architecture, featuring convolutional neural networks (gray), nonlinear activation layers (brown), a one-dimensional conversion layer (orange), and dense neural networks (green). The inputs are the magnetic predictors, and the final outputs are the predicted Z positions of the car (red). (e) Example of the ML predictions (red line) for unseen magnetic data. The ground truth measured by the camera indicates the parking levels (dashed green line), while the traveling positions are interpolated by a simple linear approximation (teal line).

is environmental magnetic noise from nearby electronic equipment and wiring, in the order of few nT, and a much larger contribution from urban electrical trams operating continuously, located approximately 50 m away from the magnetometer. These vehicles are ground cars with overhead current-carrying cables that generate magnetic fields, providing $1/f$-like random magnetic field fluctuations ranging from 70 to 250 nT.

In order to detect full-vector magnetic signals, we use a single-crystal diamond substrate containing a dense ensemble (approximately 4.5 ppm) of NV defects [Fig. 1(b)], with an average sensitivity of 400 $\mathrm{pT}/\sqrt{\mathrm{Hz}}$ per Cartesian axis (see Supplemental Material [30]). Considering the approximately 3 s timescale of the elevator system, we set a 10-Hz sampling rate to ensure a time resolution ($dt = 0.1$ s) for tracking magnetic field variations.

As the ML algorithm needs to be trained and evaluated with Z-position reference targets, it is essential to record this "ground truth" information independently, and we want to use a procedure as simple as possible so our tracking application remains flexible and can be adapted to other scenarios. Following these criteria, we simply place a camera in front of the elevator doors to record its display [Fig. 1(a)] and correlate it in time with the magnetic measurements. As the display only indicates the parking level if the elevator is parked or the traveling direction and a rough level estimation when the elevator is moving,

we can confidently determine only the time intervals in which the elevator is parked. For the complete Z trajectory, we interpolate the parking positions with a linear approximation during the traveling events, which gives a good estimate of the position. As we will discuss later, the tracking precision can be improved with more sophisticated methods to record the elevator's movement, such as measuring its acceleration as a function of time and using a physical model to reconstruct the Z position, but this approach is against the original intention of employing minimal information.

Using both sets of magnetic and Z-position measurements, we design a ML algorithm with a deep-learning supervised architecture, in which the full-vector magnetic fields $(B_X, B_Y, B_Z)(t)$ are encoded into 3-channel input predictors and the elevator position $Z(t)$ as target values [Fig. 1(c)]. During training, the internal parameters of the ML architecture are tuned to minimize the difference between the predicted positions $Z_{\text{pred}}(t)$ and the ground truth [Fig. 1(d)]. Once this process is completed, the ML algorithm can predict the elevator position in real time for magnetic measurements [Fig. 1(e)].

To assess the ML predictions' accuracy, we define two different metrics: the parking accuracy A_{park}, which takes into account only those intervals in which the elevator is parked (ground truth); and the tracking accuracy A_{track}, which uses the entire trajectory, combining both the parking intervals and the linear interpolations. The former metric is the most reliable, as it is based on the ground truth, while the second metric is built on top of a simple approximation for the elevator's movement, which gives a good estimation of the real trajectory. For any metric, a single prediction P at a time t is considered correct if the absolute difference between the predicted position $Z_{\text{pred}}(t)$ and the reference position $Z_{\text{ref}}(t)$ (ground truth or interpolated values) is smaller than a fixed position tolerance Z_{tol}:

$$P(t) = \begin{cases} 1, & \text{if } |Z_{\text{true}}(t) - Z_{\text{ref}}(t)| \le Z_{\text{tol}} \\ 0, & \text{if } |Z_{\text{true}}(t) - Z_{\text{ref}}(t)| > Z_{\text{tol.}} \end{cases} \tag{1}$$

Both metrics A_{park} and A_{track}, represented as A, are calculated once the ML algorithm is trained by using a testing dataset with unseen data and calculating the ratio between the correct and total number N of predictions:

$$A = \frac{1}{N}\sum_{i}^{N} P(t_i). \tag{2}$$

B. Magnetic measurements in different frames

The quantum diamond magnetometer platform utilized in this work is displayed in Fig. 2(a). A pair of temperature-compensated samarium cobalt ring magnets is used to generate a bias magnetic field, $\vec{B}_0$. Its projection along each NV axes is shown in Fig. 2(c), resulting in the optically detected magnetic resonance (ODMR) spectrum in Fig. 2(b). Additional magnetic field variations experienced by the NV ensemble will shift the resonance frequencies in the ODMR spectrum. To enable simultaneous measurement of all Cartesian components of a dynamically changing magnetic field, we implement a measurement protocol based on Ref. [25] that simultaneously addresses four of the eight resonance frequencies. A full breakdown of the calibration routine and measurement protocol is provided in the Supplemental Material [30].

An example of the elevator's Z position for a short time span is presented in Fig. 2(d), correlated with the magnetic signals (B_X, B_Y, B_Z) projected in the laboratory frame in Fig. 2(e) (top). In this frame, where the Z axis is aligned with the elevator's axis and the XY plane is squared with the room, there is a clear correlation between the B_X component and the elevator's movement, while the other projections (B_Y, B_Z) are much noisier. This noise directionality originates in the magnetic signals coming from the electrical trams, which exhibit weak fluctuations in the X magnetic axis (up to 70 nT) and strong variations in the YZ plane (up to 250 nT), (see Supplemental Material [30]). As we did not have any prior information about the magnetic behavior of our system, our choice of Cartesian projections XYZ for the magnetic fields fortuitously allowed us to easily identify a correlation between magnetic signals and the elevator's position. However, we may have chosen any other frame $X'Y'Z'$ in which the trams' signal is distributed across all magnetic axes evenly, obscuring this correlation. An example of this scenario is shown in Fig. 2(e) (bottom), where the magnetic data are rotated to an arbitrary frame (see Methods).

From a human perspective, the projection of the magnetic signals plays an important role for data interpretation and a bad choice such as the last example may hinder the analysis. Notice, however, that the full-vector magnetic information recorded by the magnetometer is equivalent in any frame. This situation is a good example of a real-world scenario, and our challenge is to incorporate it into our object-monitoring application, so the ML learning algorithm can work with equal performance given data projected in any frame. Consequently, we introduce the concept of "minimum accuracy" A_{min} for performance evaluation, in which a single ML architecture is trained and evaluated in different frames $F(\theta, \varphi, \alpha)$, defined by the polar θ and azimuth φ spherical coordinates of the rotation axis, along with the rotating angle α. The final performance is then defined as the lowest accuracy result:

$$A_{\text{min}} = \min_{F} A|_F, \tag{3}$$

where $A|_F$ is the accuracy (as defined in Eq. (2) evaluated at a particular frame $F(\theta, \varphi, \alpha)$.

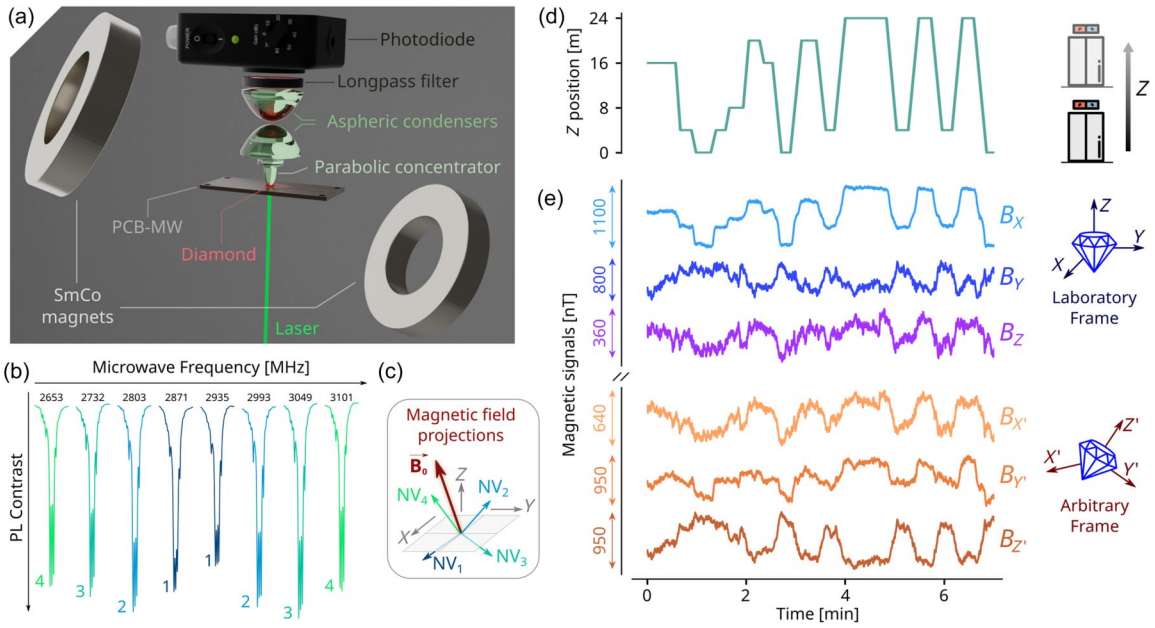


FIG. 2. Magnetic measurements in different frames. (a) Quantum diamond magnetometry platform, with details about the optical and electronic elements, along with the external magnets that provide a constant bias field $\vec{B}_0$. (b) Optically detected magnetic resonance spectrum showing the eight resonant frequencies for the ensemble of nitrogen-vacancy (NV) defects, numbered from 1 to 4 according to the NV axes. (c) Geometry for the NV axes described in the laboratory frame, along with the $\vec{B}_0$ direction. (d) Independently recorded positions of the elevator's car by filming the information display. The horizontal segments are the parking intervals (ground truth), connected with each other by linear interpolations. Parking levels are spaced approximately 4 m apart. (e) Magnetic measurements projected into the laboratory frame (top), with the Z-axis aligned with the elevator's axis, and the XY plane squared with the room. The environmental noise affects mainly the B_Y and B_Z components, leaving a relatively clean B_X axis that exhibits a clear correlation with the elevator's position. The same data are shown in a different frame (bottom), where the noise is coupled to all magnetic axes evenly, obscuring the correlation.

C. Machine-learning architecture

Considering the particular conditions of our object-monitoring challenge, but also taking into account future adaptations to similar applications, we have studied many ML algorithms that share the following principles: supervised learning, deep-learning structure, and a regression task.

In supervised learning, the ML algorithm learns from an experimental dataset containing predictors (magnetic signals) and targets (the elevator position). During training, the algorithm makes predictions based on the input and compares them with the target values. Then, it adjusts its internal parameters to reduce the difference next time, iterating this process over many cycles to improve the accuracy progressively. After training, the ML algorithm is able to predict the elevator position from magnetic measurements without any target references.

A deep-learning structure means that the ML algorithm has multiple hidden layers between the input predictors and the final output, enhancing its ability to analyze complex patterns [31,32]. This type of architecture works well with our extensive magnetic data: just 1-h of measurements at 10 Hz generates 36 000 datapoints for each magnetic component. As we will explain later, our ML algorithms perform better when they use the recent magnetic history to predict a single elevator position.

Finally, a regression task implies that the ML algorithm predicts continuous output values, in this case, the Z position of the elevator. As mentioned earlier, our ground truth measurements of the elevator's trajectory are those intervals in which the elevator is parked, while the traveling events are approximated by linear interpolations. The former information suggests a classification task, training the ML algorithm to make categorical predictions such as "Parked at Level X" or "Traveling up/down." However, this approach introduces large uncertainties for the traveling events, as it leaves intermediate positions undefined, which can extend over long durations for the longer flights. Conversely, the continuous predictions provide a good approximation of the elevator's position throughout its entire trajectory, even if the exact location remains uncertain.

An overview of the ML training process is displayed in Fig. 3. In our ML design, at any given time t, a predictor is defined as a "time window" containing magnetic measurements (B_X, B_Y, B_Z) over a fixed interval Δt [shaded area in Fig. 3(a)]. The corresponding target is the elevator position $Z_{\text{true}}(t)$, meaning that predictions are made in real

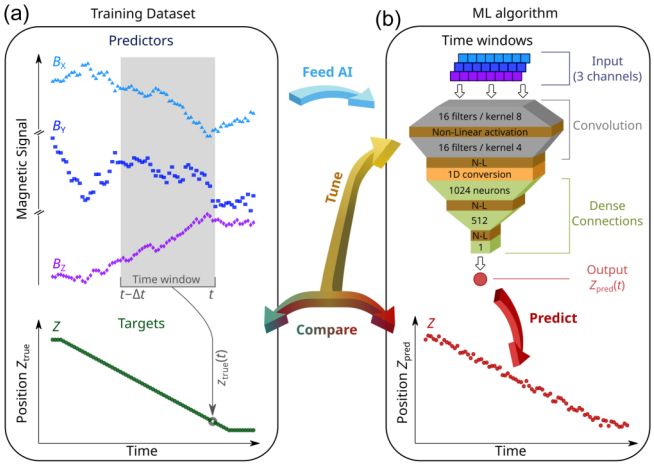


FIG. 3. Training process for the machine-learning (ML) algorithm. (Aa) Training dataset, including magnetic signals as predictors and the elevator's positions as targets. At any time t, a predictor is defined as a "time window" of duration Δt, containing the most recent magnetic history. The corresponding target is the ground truth position $Z_{\text{true}}(t)$. (b) ML architecture. From top to bottom: the input layer allocates the input time window in a three-channel matrix, passing this information into a set of convolutional neural networks, characterized by their number of filters and kernel size, along with nonlinear activation (N-L) functions. The multidimensional output, corresponding to the different filters, is flattened into a one-dimensional (1D) vector by a conversion layer and then input into a series of dense layers, characterized by the number of neurons, and N-L functions. The final output is a single value representing the predicted position of the elevator $Z_{\text{pred}}(t)$ for the input time window. During a training cycle, the entire set of predictions is compared to the reference targets, computing the "loss" score as the average value of the absolute difference. The ML internal parameters are then tuned, aiming to minimize the loss for the next cycle. This process is repeated for a fixed number of cycles or until an early stop criterion is met.

time by taking into account the most recent magnetic history. Since the ML algorithm is designed to produce continuous outputs, we use all Z positions as ground truth during training, including both parking and traveling intervals. In contrast, during the evaluation process, we do distinguish between the ground truth parking positions and the interpolated approximations by using the A_{park} and A_{track} metrics defined previously.

The architecture of our ML algorithm is illustrated in Fig. 3(b). From top to bottom, the structure begins with the input layer, which allocates a single predictor: a $3 \times N$ matrix that represents the time window with the $(B_{\text{X}}, B_{\text{Y}}, B_{\text{Z}})$ components in separate channels (rows), and N points given by the interval's length Δt and the time resolution $dt = 0.1$ s: $N = \Delta t/dt$. This information is processed by a block of one-dimensional CNN and nonlinear activation (N-L) functions, which explore long- and short-range correlations within the data and extract the most relevant patterns for the next layers. Our ML algorithm works best with two CNN, each of them having 16 filters (convolutional matrices that analyze the input data), with kernel sizes 8 and 4 for the first and second CNN, respectively. During the convolutional process, the original data are expanded into multiple dimensions (as many as the number of filters), which are then flattened into a single vector by a one-dimensional (1D) conversion layer. This format is required by the next processing block: a series of dense (or fully connected) layers and N-L functions. Each dense layer is characterized by a number of neurons n, which are processing units that receive all input datapoints, multiply them by internal weights and add biases, and finally return an output vector with size n. In our architecture, we progressively downsize the information with dense layers of sizes $n = 1024$, 512, and 1, until we get a final value as output: the predicted position of the elevator $Z_{\text{pred}}(t)$ for the input time window.

During training, the ML algorithm initializes its internal parameters randomly and updates them every cycle (or epoch) to improve the future predictions. In each cycle, the complete set of predictors are input into the algorithm and the output predictions $Z_{\text{pred}}(t)$ are compared to the ground truth $Z_{\text{true}}(t)$, computing a "loss" score equal to the average value for the absolute difference $|Z_{\text{pred}}(t) - Z_{\text{true}}(t)|$. The algorithm consequently modifies its internal parameters with a back-propagation method [32] trying to minimize the loss for the next epoch [see arrows connecting [Figs. 3(a) and 3(b)]. This process is iterated for a fixed number of cycles or until an early stop criterion is met (see Methods).

In the evaluation process, the ML architecture is fed with an unseen set of time windows, producing the output predictions $Z_{\text{pred}}(t)$ but not updating the internal parameters. The predictions are then compared with the reference values $Z_{\text{true}}(t)$, and the accuracies A_{park} and A_{track} are calculated according to Eq. (2). The entire training and evaluation procedures are repeated for different frames, to ensure a robust ML performance, and the final accuracies $A_{\text{park, min}}$ and $A_{\text{track, min}}$ are computed using Eq. (3).

D. Optimization of hyperparameters

In any ML algorithm, there are two kinds of parameters: internal parameters, which are tuned automatically during the training process; and hyperparameters, which define global structures and require human input. Examples of the later are the type and number of layers within the ML architecture, the type of nonlinear activation function to be used or the number of points in a time window. By optimizing hyperparameters, we can significantly boost the ML performance. In this section, we describe a series of optimization stages that we carried out to find our best performing ML algorithm.

As the general condition for each optimization stage, we studied different options for a single hyperparameter, or a

reduced combination of them, leaving the other hyperparameters fixed. For each option, we evaluated several possibilities for ML architectures according to the metric $A_{\text{track, min}}$ using a tolerance $Z_{\text{tol}} = 1$ m and several frames for the magnetic data. Among all the results for a single option, we chose the lowest accuracy as the final score. As a reference benchmark, we set a minimum accuracy threshold of 80%, meaning that, on average, eight of ten predictions are correct every second.

The first optimization stage, which addresses the full-vector nature of the measurements, is shown in Fig. 4(a): do we need all (B_X, B_Y, B_Z) magnetic components for accurate predictions, or can we use just the scalar field $|\vec{B}|$

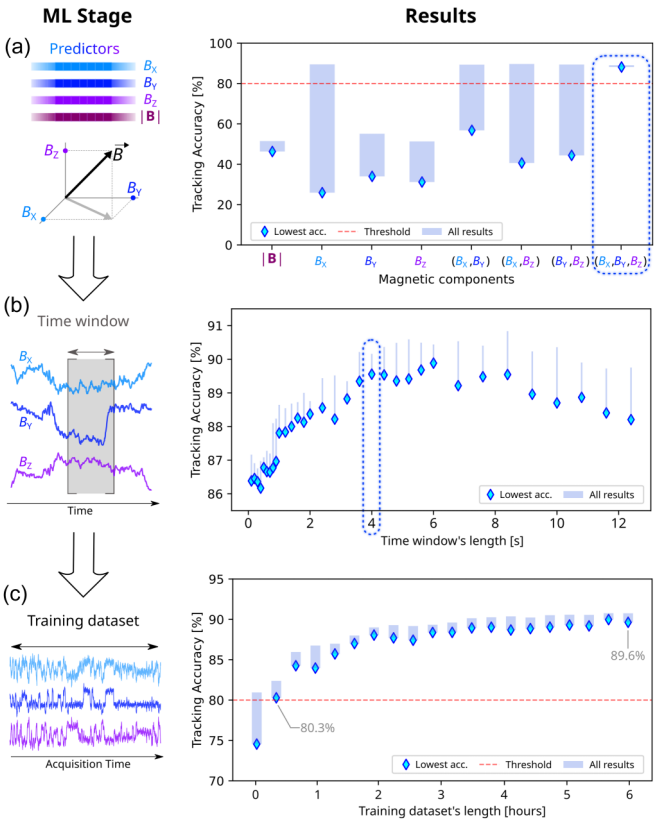


FIG. 4. Optimizing the machine-learning hyperparameters. In progressive stages, multiple options for a single hyperparameter are studied, training several algorithms and computing the tracking accuracy with a 1-m tolerance. While the complete range of results are displayed (shaded bars), only the lowest results (blue diamonds) count as the final score. We set a reference benchmark of 80% (red dashed line). (a) First stage: combinations of magnetic components used as input predictors, from which the full-vector option is selected (dashed rectangle). (b) Second stage: duration of time windows. There is an optimal range from 4 to 6 s, from which we select the shorter time (dashed rectangle), favoring a simpler computational structure of the ML algorithm. (c) Final stage: impact of the training dataset size. Larger sizes improve the tracking accuracy, up to 89.6% when using the full dataset (6 h). If only an 80% accuracy is required, using just 0.5 h of measurements is enough.

or a single magnetic projection? This decision has important implications for the complexity of the ML algorithm, as well as for the types of sensing platforms that can be utilized in this application. For each hyperparameter option, the final score is represented by a blue diamond, while the full range of results is shown as a shaded bar. Noticeably, only the full-vector predictors (B_X, B_Y, B_Z) overcome the 80% accuracy threshold (red dashed line), showing little variation across all results. In contrast, all other magnetic component combinations exhibit low scores (less than 50%), with great variability in results.

The importance of having the complete magnetic information in the input data becomes clear when analyzing the large spread in accuracy results. For some frames, one or two magnetic projections may capture the essential features of the signal, which explains the higher end for the accuracy results in options such as the single B_X. However, there are other frames where the reduced information becomes insufficient, hence showing a low performance. The scalar field option $|\mathbf{B}|$, on the other hand, is invariant to any frame representation at the expense of losing directional information, which negatively impacts the ML performance. Conversely, the full-vector data (B_X, B_Y, B_Z) always encode the complete information, irrespectively of the chosen frame, and achieves consistently high accuracy. Consequently, we choose this option for the subsequent stages.

The second optimization stage studies the duration of the time windows: what is the optimal length for accurate predictions? The results are shown in Fig. 4(b), where we notice that all options score above 86%, even the single-point time window (left-most result), which is a simple magnetic vector. However, we observe an optimal range approximately from 4 to 6 s (equivalent to 40 to 60 points, given our 10-Hz measurement rate). We speculate that too short time windows do not unlock the full potential of the convolutional layers, while too long time windows provide excessive information [33]. As shorter time windows imply less internal parameters for the ML algorithm, which is translated into faster training and less chances to overfit the data, we choose a 4-s duration as the optimal length.

Further optimization stages are described in the Methods section, resulting in the ML architecture shown in Fig. 3(b). Fine details of the algorithm include the type of nonlinear activation functions, defined as hyperbolic tangent ("tanh") in all cases, the global Adaptive Moment Estimation ("adam") optimizer and the 0.0001 learning rate.

For the best ML architecture, we finally evaluate the effect of the training dataset size (measured in recording hours) on the tracking accuracy performance, as illustrated in Fig. 4(c). As expected, the larger the training dataset the better the accuracy, reaching an optimal value of 89.6% with 6 h of measurements. We also notice that the

performance is reaching a limit at this point, suggesting that while the dataset size is critical for training the ML algorithm, it is not the only factor. Other contributions are likely related to the uncertainty in the magnetic measurements, set by the magnetic noise floor, unexpected events and imperfections of the target object. In other words, we do not expect perfect predictions in a real-world scenario. On the other hand, if the monitoring application requires 80% tracking accuracy, recording just half an hour of measurements is enough to train the ML algorithm, which is a reasonable time if the access to ground truth measurements is limited or if there is a need for periodical recalibrations.

As a final comment, we studied the possibility of artificially augmenting the training dataset to reduce the measurements time even further. For this purpose, we reduced the original training dataset to a fraction and then generated similar copies which introduced random variations for the predictors, while the ground truth targets remained unchanged. When the ML algorithms were trained with the augmented predictors, the results showed that data augmentation, by any amount, did not improve the performance, and in most cases made it worse (see Supplemental Material [30]). This result reinforces the advantage of training the ML algorithm exclusively on experimental data: synthetic information cannot be easily used as a substitute of the real measurements. If carefully engineered, artificial data may emulate reality, but that would require prior knowledge about the system, and this is precisely the limitation that we are overcoming with our method.

E. Final performance

After selecting the optimal hyperparameters for the ML architecture, we compute the final performance of the algorithm using the testing dataset, which spans 6 h of measurements, and a collection of 65 randomly generated frames for the magnetic data projection. We compute both the tracking and parking accuracies according to Eqs. (1)–(3), using a position tolerance $Z_{\mathrm{tol}} = 1$ m.

The input magnetic fields from a 1-h segment of the testing dataset are shown in Fig. 5(a), for a particular frame in which there are not clean magnetic axes, making the data analysis apparently difficult from a human perspective. After these data are fed into the ML algorithm, the predicted Z positions are compared with the reference ground truth, see Fig. 5(b). The prediction accuracies, when computed in the entire testing dataset and the 65 frames, are $A_{\mathrm{track,\,min}} = 88.9\%$ (general tracking) and $A_{\mathrm{park,\,min}} = 94.9\%$ (only parking positions), as displayed in Fig. 5(c). Following this result, we demonstrate that our ML method can reliably monitor the position of the elevator in real time, with an average of nine out of ten predictions correct every second.

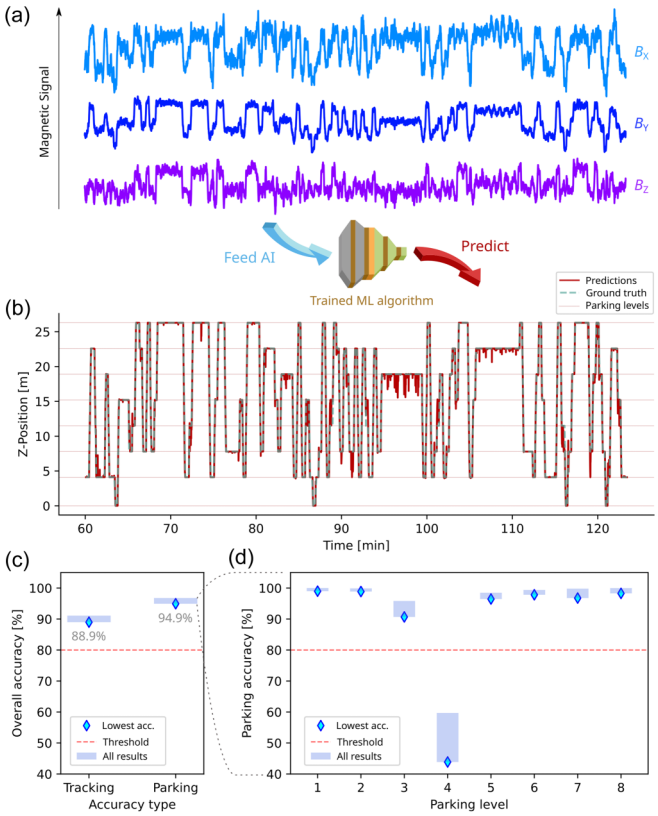


FIG. 5. Final performance for the object-monitoring machine-learning application. (a) Input magnetic predictors from a segment of the testing dataset, in a particular frame where the environmental noise is distributed across all magnetic components, making the data analysis apparently difficult from a human perspective. (b) Predictions for the target's position (red solid line), compared with the ground truth (teal dashed line). Parking levels are indicated with faint horizontal lines. (c) Tracking and parking accuracies evaluated in the entire testing dataset (6 h) across 65 randomly generated frames for the magnetic data projection. The final performance is defined as the lowest result within these frames. (d) Parking accuracy examined for each parking level, demonstrating that predictions are more than 90% accurate except for those related to level 4.

In addition to the global results, the parking accuracy is examined in the different parking levels in Fig. 5(d), meaning that the predictions are separated in groups associated to each (ground truth) parking level. This analysis shows that the parking predictions are very accurate ($A_{\mathrm{park,\,min}} > 90\%$) for all parking levels except for level 4 [$Z = 11.5$ m in Fig. 5(b)]. We speculate that at this position the elevator's car and counterweight are close, in a geometric configuration that produces a complex magnetic signal which is easily obscured by the noise and confused by the ML algorithm with the neighboring positions (see Supplemental Material [30]). In fact, if we increase the position tolerance to $Z_{\mathrm{tol}} = 4$ m, the prediction's accuracy becomes higher than 99% for all parking levels (see Supplemental Material [30]), demonstrating

that the incorrect predictions in the previous case were assigned to the adjacent parking levels or in-between, and not anywhere further.

Throughout all previous analyses, we have studied the performance of the ML algorithm related to a fixed tolerance value $Z_{\text{tol}} = 1$ m, which is small compared to the parking levels separation (approximately 4 m), and allows for a certain parking level identification. However, we can extend the analysis and explore the tracking resolution of our ML method for the general tracking task, in terms of position tolerance. By following the same procedure described at the beginning of this section but for a continuous Z-position tolerance, we compute the tracking accuracy $A_{\text{track, min}}$ as a function of Z_{tol}, as displayed in Fig. 6(a). For a minimum accuracy of 80%, the tolerance can be set as low as 30 cm, demonstrating the precision and robustness of our monitoring application. As previously described in Fig. 5(a), the tracking accuracy for $Z_{\text{tol}} = 1$ m is 88.9%, while the ultimate limit of $Z_{\text{tol}} = 2$ m, which still guarantees a reliable identification of the closest parking level, increases the performance up to $A_{\text{track, min}} = 95.0\%$.

Finally, we show a complete map of the tracking accuracy A_{track} for each frame $F(\theta, \varphi, \alpha)$ in Fig. 6(b), for a 1-m position tolerance. The map projects the polar θ and azimuth φ coordinates of the different rotating axes (in the laboratory frame), while the rotating angle is fixed to $\alpha = 90°$ (see Methods), with each point representing the prediction results, color-coded by the accuracy value. Across all frames, the accuracy variability is less than 3%, demonstrating the robustness of the ML algorithm. In addition, as there is no clear spatial pattern for the accuracy distribution, we speculate that this small spread is most probably coming from the effect of the random initialization of the ML internal parameters rather than from a geometrical configuration.

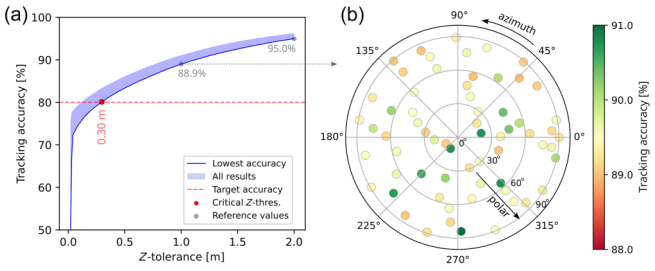


FIG. 6. Tracking accuracy as a function of the Z-position tolerance. (a) The monitoring performance of the machine-learning algorithm is studied for a continuous range of tolerance values, using a collection of 65 randomly generated frames for the magnetic data projection. The final score is defined as the lowest accuracy across all frames (solid blue line), while the full range of results is displayed as a shaded area. For an accuracy threshold of 80%, the tolerance can be as low as 30 cm. (b) Distribution of tracking accuracy results (color-coded) in the different frames using a 1-m position tolerance, projected according to the polar and azimuth coordinates that describe each frame.

III. DISCUSSION

In this work, we have developed a machine-learning algorithm trained exclusively on experimental data, obtained by a quantum diamond magnetometer, to monitor in real time the activity of a moving magnetic target within a noisy environment. The ML performance was classified into two different accuracies: general tracking $A_{\text{track, min}}$ and parking position $A_{\text{park, min}}$. When using a 1-m position tolerance to evaluate the correct predictions, the performance is $A_{\text{track, min}} = 88.9\%$ and $A_{\text{park, min}} = 94.9\%$. If only an 80% accuracy is required for the general tracking task, then the position tolerance can be as low as 30 cm.

While machine-learning has been applied to similar monitoring challenges, this is the first reported method for real-time magnetic tracking based on pure experimental training, without the need of building intricate physical models or simulations. As a matter of fact, the ML method can be improved with more information, models and simulations, by augmenting the training dataset or providing rules for the predictions. For example, the predictions in Fig. 5(b) show unstable positions at some parking intervals, implying that the elevator is going up and down too fast. If we incorporated physical-informed rules to the deep learning structure, such as penalizing the algorithm for predicting too large "jumps" in position, the accuracy would increase. Although any improvements are beneficial, the simple ML architecture that we have developed in this work follows a very easy implementation with a demonstrated high performance.

An example of enhancing the ML performance by adding more information is to provide a better interpolation method between parking intervals. Our choice of linear approximation is not entirely realistic as it implies instant acceleration at the beginning and end of each flight, although it provides a good position estimation. Alternatively, we have experimentally measured the elevator's acceleration profile and built a physical model for the position as a function of time, see the Supplemental Material [30] for more details. When the ML algorithm is trained with this type of interpolation, by the end of the final optimization stage, the tracking accuracy slightly improves from the previous 89.6% to 91.1%. For this particular scenario, a 1.5% gain probably does not justify the effort of acquiring the additional acceleration measurements and physical model. However, the takeaway message is that the basic ML algorithm can be improved as much as the user wants for their particular application, at the expense of more knowledge and/or computational design, but the baseline performance is already high.

The applicability of this research can be extended to a wide range of situations where magnetic objects move within a relatively controlled (though complex)

environment, meaning that the external perturbations are limited. This is the critical factor for the supervised ML algorithm, as it must be trained with experimental data and account for a reasonable number of outcomes. In a continuously changing environment, where significant and distinct magnetic perturbations occur frequently, the training dataset cannot cover all possibilities. Conversely, in a controlled environment where magnetic perturbations remain within some boundaries, like the strong but steady noise from the distant electrical trams in our work, the ML algorithm can learn general trends and make accurate predictions, just from experimental data. For example, we could apply our method to monitor magnetic objects in underground mines or tunnels (where the magnetic environment is relatively free from external factors), such as vehicles or miners carrying magnetic beacons. The more complex the environment and target signals, the larger the required training dataset. Nevertheless, training the ML algorithm is a one-time investment, and it can be a solution for problems where theoretical models or simulations are not available.

Our application integrates a quantum sensing platform with a machine-learning algorithm to solve a problem that is hardly approachable from a conventional point of view. The adaptability of our approach, where the magnetic sensor and ML design can be customized to the specific requirements of the problem, makes it a versatile and promising solution for a wide range of object-monitoring applications based just on experimental measurements.

ACKNOWLEDGMENTS

We thank Adam Silvester and Liam T. Hall for fruitful discussions. This work was financially supported by the Australian Research Council (ARC) Centre of Excellence Scheme (Grants No. CE170100012 and No. CE230100021). D. A. S. and L. T. H. acknowledge support from the National Intelligence and Security Discovery Research Grants Program (NS220100071). This work was partially supported by Defence through the Next Generation Technologies Fund (NGTF) and managed under the Advanced Strategic Capabilities Accelerator. Additional ARC funding support was provided through (IM240100073) and by The University of Melbourne's Research Computing Services and the Petascale Campus Initiative.

D. A. S., A. Sa., B. A. G., A. G., and L. C. L. H. contributed to conceptualization. F. M. contributed to the machine-learning methodology. C. T. K. L. contributed to the experimental methodology. F. M., C. T. K. L., D. A. S., A. Si., A. Sa., B. A. G., A. G., and L. C. L. H. contributed to the analysis. F. M. and C. T. K. L. contributed to visualization. D. A. S., L. C. L. H., B. A. G., and A. G. contributed to funding acquisition. D. A. S. and L. C. L. H. contributed to supervision. F. M., C. T. K. L., and D. A. S. contributed to writing the original draft. F. M., C. T. K. L., D. A. S., A. Si., A. Sa., B. A. G., A. G., and L. C. L. H. contributed to writing, reviewing, and editing the manuscript.

D. A. S, A. Sa, B. A. G and A. S. declare that they are involved in the development and commercialization of technology related to this article through a start-up company Phasor Quantum. D. A. S, A. Sa, and B. A. G are co-founders and equity holders in Phasor Quantum. Phasor Quantum did not participate in the design, data collection, analysis, interpretation, manuscript preparation, or the decision to publish. Appropriate measures were taken to support the independence and transparency of the research. All relevant interests have been disclosed, and the authors declare no other conflicts of interest relevant to this work.

DATA AVAILABILITY

All of the code used and the data presented in this work are available in the GitHub repository at [34].

APPENDIX A: QUANTUM DESCRIPTION OF NITROGEN-VACANCY DEFECTS

The NV defect is a substitutional nitrogen atom adjacent to a vacancy aligned along the $\langle 111 \rangle$ crystallographic axes of the diamond lattice, which behaves as a spin-1 system. The ground state spin triplet includes a nonmagnetic $m_S = 0$ state and two energy-degenerate $m_S \pm 1$ states which are separated from the $m_S = 0$ state by a zero field splitting of 2.87 GHz. The full Hamiltonian describing the energy levels can be found in the Supplemental Material [30]. When exposed to an external magnetic field, the ground state magnetic levels $m_S = \pm 1$ split due to the Zeeman interaction. The energy level splitting is proportional to the magnetic field projection along the NV axis, meaning that each NV defect accounts for specific magnetic information according to its orientation in the crystal lattice. Any external magnetic field vector can be reconstructed by measuring the magnetic field projections along the four possible NV axes within a dense NV ensemble in the diamond.

APPENDIX B: EXPERIMENTAL SETUP

The quantum diamond magnetometry setup is described below in three main sections concerning the optical, electronics and lock-in setups.

1. Optical setup

The magnetic sensing unit is a $3 \times 3 \times 0.5$ mm^3 diamond chip (DNV-B14) from Element-6 with a typical NV concentration of 4.5 ppm. A high power laser (Laser Quantum Opus 532, 6 W) is used to illuminate the

diamond, at a power of 500 mW during normal operating conditions. The laser output passes through an optical isolator (Thorlabs IO-5-532-HP), half waveplate (Thorlabs WPH10M-532), beam sampler (thorlabs BSF10-A), two steering mirrors (Edmund No. 38-893) and a $f = 500$ mm focusing lens (Thorlabs AC254-500-A-ML). A final mirror is then used to direct the focused laser beam through a 1-mm diameter hole in the printed circuit board (PCB) with the embedded microwave (MW) antenna design [35]. With approximately 320 mW of green excitation measured just before illuminating the diamond, 10.6 mW of NV photoluminscence (PL) is measured.

PL from the diamond is collected using a compound parabolic concentrator (Edmund No. 17-709), followed by two back-to-back aspheric lenses condenser (Thorlabs ACL25416U-B), a 635-nm longpass filter (Semrock BLP01-635R-25) and a large area photodetector (Thorlabs PDA36A2). A small portion of the laser output picked off using the beam sampler is directed toward a second photodetector (Thorlabs PDA36A2) for common-mode rejection of laser intensity noise after passing through a continuously variable neutral density filter (Thorlabs NDC-50 C-2M) and a 532-nm bandpass filter (Semrock FF01-524/24-25). The common-mode rejected photodetector signal is then simultaneously digitized by eight ac-coupled analog-to-digital converters (ADCs) at a sampling rate of 202.8 kSa/s (4× National Instruments (NI) PXI-4461 modules housed within a NI PXIe-1062Q chassis) and averaged in software to reduce quantization noise.

2. Electronics setup

Four MW synthesizers (2× Agilent E4432B, 2× Agilent E4433B) are used to generate four frequency-modulated (FM) multitone signals to address four of the eight NV resonances and their three hyperfine spin transitions spaced ±2.158 MHz apart. The FM square waveforms are generated by four digital-to-analog converters at a sampling rate of 202.8 kSa/s from four analog outputs on two of the NI PXI-4461 modules. A 10-MHz oven-controlled crystal oscillation clock source is used to synchronize all MW synthesizers and modules housed within the NI chassis by connecting in a daisy-chain manner.

The MW signal from each synthesizer passes through a dc block (Minicircuits BLK-89-S+) and an isolator (Microwave Associates MZG-3000) before being combined with each other using three power combiners (Minicircuits ZX10Q-2-34-S+). An amplifier (Minicircuits ZHL-5W-422+) is then used to amplify the combined MW signal before sending the signal to the embedded PCB MW antenna.

3. Lock-in setup

The lock-in demodulation of the common-mode rejected, averaged, and frequency-modulated PL response is performed in software. First, a tenth-order Butterworth highpass filter with a cutoff frequency at 1500 Hz is applied. The filtered signal is then mixed with the four square reference waveforms individually. Each demodulated trace is then lowpass filtered (tenth-order Butterworth, 53 Hz), removing the $2f_i$ second harmonic component. Finally, the demodulated and filtered traces are decimated and averaged by a factor of 20 280, downsampling from 202.8 kSa/s to 10 Sa/s.

APPENDIX C: TRAINING AND TESTING DATASETS

The total set of measurements, including magnetic signals and the elevator car's position, comprises 12 h acquired in five different segments. We only include daytime measurements, when the elevator is actively used. All timeseries for magnetic and position measurements can be found in the Supplemental Material [30], along with a histogram for the B_X component in the laboratory frame where eight peaks can be resolved, corresponding to the eight parking levels within the building (Supplemental Material [30]). The complete dataset spans 426 496 points for each magnetic component and target position, which is split equally into the training dataset (211 756 points) and testing dataset (214 740 points).

APPENDIX D: FRAMES FOR MAGNETIC DATA

The original magnetic data, projected into the laboratory frame XYZ, can be converted to any other frame $X'Y'Z'$ by a rotational transformation around an axis $\hat{n}$ by an angle α. We identify a rotated frame $F(\theta, \varphi, \alpha)$ by the polar θ and azimuth φ spherical angles (expressed in the laboratory coordinates) of the rotating axis $\hat{n}$ and a rotating angle α. The mathematical description of the rotating transformations can be found in Supplemental Material [30].

In the main Secs. II B and II C, the ML algorithms are trained and evaluated in the following frames: laboratory [Fig. 2(e) top], equivalent to $F(0°, 0°, 0°)$; $F(67°, 59°, 120°)$ where the noise is partially distributed; and $F(58°, 61°, 90°)$ where the noise is evenly distributed across all magnetic components [Fig. 2(e) bottom]. In the main Sec. II E, multiple rotational frames are generated by randomly selecting the spherical coordinates within ranges $\theta \in (0°, 90°]$ and $\varphi \in (0°, 360°]$, and fixing the rotating angle to $\alpha = 90°$.

APPENDIX E: DATA NORMALIZATION AND TIME WINDOWING

As machine-learning algorithms perform better with normalized input data [36,37], we apply a normalization process to our magnetic datasets. This process converts the values of each magnetic component into a dimensionless scale, centered at zero and with unitary amplitude. First, we account for the Earth's background magnetic field, which remains effectively constant in our measurement

timescale (see Supplemental Material [30]) by subtracting the mean value in each magnetic component. Next, we identify the maximum scalar value $B_{\max}$ within the training dataset and define the normalization factor as $B_{\text{norm}} = B_{\max}/\sqrt{3}$. Finally, all the data (both training and testing datasets) are divided by B_{norm}, resulting in dimensionless datasets with zero mean value and variance smaller than 1. In the main text, the magnetic data shown in figures are already normalized, with the exception of Fig. 2(e). There, we show the original magnetic variations for each component, after subtracting the Earth's magnetic field.

Following data normalization, the magnetic information is processed into time windows with a fixed duration Δt, which are the input units for the machine-learning algorithm. For each time t associated with the elevator's car position $Z(t)$, the corresponding time window contains the magnetic information within the time interval $[t - \Delta t, t]$. These data are processed as a $N_{\text{channels}} \times N_{\text{points}}$ matrix, where N_{channels} is the number of magnetic components that are used as input data, e.g., $N_{\text{channels}} = 3$ for the full-vector option (B_X, B_Y, B_Z), and N_{points} is the number of measurements within the interval $[t - \Delta t, t]$.

APPENDIX F: GENERAL ARCHITECTURE FOR ALL ML ALGORITHMS

The ML architecture shown in Fig. 3(b) is structured in a series of layers which are representative of all the algorithms we have explored. Although the hyperparameters of each layer may change in a particular design, we always use the following sequence: (i) input layer (predictors); (ii) set of one-dimensional convolutional neural networks (1D-CNN) layers, nonlinear activation functions (N-L) and optional pooling layers; (iii) one-dimensional conversion layer; (iv) set of dense layers, N-L and optional dropout layers, always including a final dense layer with a single neuron; and (v) output layer, a single value representing the position of the elevator's car (target). Notice that the architecture illustrated in the main text does not include any pooling or dropout layers, which are optional. A detailed explanation for each of the architecture elements can be found in Supplemental Material [30].

In addition to the sequence of layers, all the ML algorithms are characterized by the following global hyperparameters: Mean absolute error as the loss function; validation fraction of 25% during training; early stop criterion of $N_{\text{early}} = 15$ training cycles without any improvement in the validation loss score; and maximum number of training cycles equal to $N_{\max} = 200$. For the final training stage, in which we use the optimized hyperparameters, we extend the early stop criterion to $N_{\text{early}} = 30$ and the maximum number of training cycles to $N_{\max} = 400$.

APPENDIX G: TRAINING STAGES FOR HYPERPARAMETERS OPTIMIZATION

Our best ML architecture is built after a sequence of six optimization stages, focused on the following hyperparameters: (i) input magnetic components; (ii) time window's length; (iii) main architecture (variations within the layers' structure); (iv) global hyperparameters; (v) fine architecture (local hyperparameters); and (vi) training dataset size. Stages 1, 2, and 6 are described in the main text (Fig. 4), while the complete description is available in Supplemental Material [30]. After the final optimization stage, we keep the best ML architecture described in the main text, which has 1 019 297 internal parameters.

In all the optimization stages, the performance of every hyperparameter option (i.e., the lowest tracking accuracy) includes the accuracy results of several ML architectures. Each architecture design was trained and evaluated in three frames for the magnetic data: $F(0°, 0°, 0°)$, $F(67°, 59°, 120°)$, and $F(58°, 61°, 90°)$; as well as three different random seeds for the initialization of internal parameters. Therefore, each shaded bar containing all accuracy scores in Fig. 4 includes nine results per ML architecture, multiplied by the number of evaluated architectures (which varies according to the optimization stage, see Supplemental Material [30]).

APPENDIX H: ML LIBRARIES AND COMPUTING RESOURCES

The programming code developed in this work was written in PYTHON 3.10 with TENSORFLOW 2.11.0. All ML training processes were carried out on a personal laptop: Lenovo ThinkPad X1 Extreme Gen 5, 12th Gen Intel® Core™ i7-12800H, equipped with a NVIDIA RTX GeForce 3050 Ti GPU. Considering the architecture of our best model, the training time for 400 epochs is approximately 40 min. Once the model is trained, the prediction time for a single time window is approximately 0.6 ms (much faster than our 10-Hz measurement rate).

[1] W. Sun, L. Dai, X. Zhang, P. Chang, and X. He, RSOD: Real-time small object detection algorithm in UAV-based traffic monitoring, Appl. Intell. **52**, 8448 (2022).
[2] R. Ke, Z. Li, S. Kim, J. Ash, Z. Cui, and Y. Wang, Real-time bidirectional traffic flow parameter estimation from aerial videos, IEEE Trans. Intell. Transp. Syst. **18**, 890 (2017).
[3] R. Kays, M. C. Crofoot, W. Jetz, and M. Wikelski, Terrestrial animal tracking as an eye on life and planet, Science **348**, aaa2478 (2015).
[4] G. Gariepy, F. Tonolini, R. Henderson, J. Leach, and D. Faccio, Detection and tracking of moving objects hidden from view, Nat. Photonics **10**, 23 (2016).
[5] H. Ghafoor and Y. Noh, An overview of next-generation underwater target detection and tracking: An integrated underwater architecture, IEEE Access **7**, 98841 (2019).

[6] Y. Jiang, W. Chen, X. Zhang, X. Zhang, and G. Yang, Real-time monitoring of underground miners' status based on mine IoT System, Sensors **24**, 739 (2024).

[7] Y. Bock and D. Melgar, Physical applications of GPS geodesy: A review, Rep. Prog. Phys. **79**, 106801 (2016).

[8] R. B. Langley, Propagation of the GPS signals, in *GPS for Geodesy*, edited by A. Kleusberg and P. J. G. Teunissen, Lecture Notes in Earth Sciences, Vol. 60 (Springer, Berlin, Heidelberg, 1996), pp. 103–140.10.1007/BFb0117680

[9] H. S. Hewawasam, M. Y. Ibrahim, G. Kahandawa, and T. A. Choudhury, Comparative study on object tracking algorithms for mobile robot navigation in GPS-denied environment, in *Proceedings of the IEEE International Conference on Industrial Technology*, Melbourne, Australia (IEEE, 2019), pp. 19–26.10.1109/ICIT.2019.8754960

[10] X. Wang, W. Li, B. Moran, B. C. Gibson, L. T. Hall, D. A. Simpson, A. N. Kealy, and A. D. Greentree, Quantum diamond magnetometry for navigation in GNSS denied environments, in *Gravity, Positioning and Reference Frames*, edited by J. T. Freymueller and L. Sánchez, Vol. 156 (Springer, Cham, 2022), pp. 87–92.10.1007/1345_2023_218

[11] J. Callmer, M. Skoglund, and F. Gustafsson, Silent localization of underwater sensors using magnetometers, EURASIP J. Adv. Signal Process. **2010**, 709318 (2010).

[12] N. Wahlström and F. Gustafsson, Magnetometer modeling and validation for tracking metallic targets, IEEE Trans. Signal Process. **62**, 545 (2014).

[13] Z. Chen, R. Chen, T. Deng, Y. Wang, W. Di, H. Luo, and T. Han, Magnetic anomaly detection using three-axis magnetoelectric sensors based on the hybridization of particle swarm optimization and simulated annealing algorithm, IEEE Sens. J. **22**, 3686 (2022).

[14] A. Soheilian, M. Ranjbaran, and M. M. Tehranchi, Position and direction tracking of a magnetic object based on an Mx-atomic magnetometer, Sci. Rep. **10**, 1294 (2020).

[15] X. Wu, S. Huang, M. Li, and Y. Deng, Vector magnetic anomaly detection via an attention mechanism deep-learning model, Appl. Sci. **11**, 11533 (2021).

[16] J. Cárdenas, C. Denis, H. Mousannif, C. Camerlynck, and N. Florsch, Magnetic anomalies characterization: Deep learning and explainability, Comput. Geosci. **169**, 105227 (2022).

[17] T. Sun, X. Wang, J. Wang, X. Yang, T. Meng, Y. Shuai, and Y. Chen, Magnetic anomaly detection of adjacent parallel pipelines using deep learning neural networks, Comput. Geosci. **159**, 104987 (2022).

[18] Y. Wang, Q. Han, G. Zhao, M. Li, D. Zhan, and Q. Li, A deep neural network based method for magnetic anomaly detection, IET Sci. Meas. Technol. **16**, 50 (2022).

[19] B. N. J. Vandavasi, H. Venkataraman, and A. R. Gidugu, Machine learning-based electro-magnetic field guided localization technique for autonomous underwater vehicle homing, Ocean Eng. **280**, 114692 (2023).

[20] X. Meng, Y. Zhang, X. Zhang, S. Jin, T. Wang, L. Jiang, L. Xiao, S. Jia, and Y. Xiao, Machine learning assisted vector atomic magnetometry, Nat. Commun. **14**, 6105 (2023).

[21] Y. Liu, Y. Wu, L. Yang, P. Zhou, J. Kuang, W. Yu, J. Wang, Z. Xu, and G. Li, A Multi-task learning for submarine cable magnetic anomaly recognition, J. Mar. Sci. Eng. **11**, 900 (2023).

[22] T.-H. Chiang, Z.-H. Sun, H.-R. Shiu, K. C..-J. Lin, and Y.-C. Tseng, Magnetic field-based localization in factories using neural network with robotic sampling, IEEE Sens. J. **20**, 13110 (2020).

[23] P. Li, M. Abdel-Aty, Q. Cai, and Z. Islam, A deep learning approach to detect real-time vehicle maneuvers based on smartphone sensors, IEEE Trans. Intell. Transp. Syst. **23**, 3148 (2022).

[24] S. C. Scholten, A. J. Healey, I. O. Robertson, G. J. Abrahams, D. A. Broadway, and J.-P. Tetienne, Widefield quantum microscopy with nitrogen-vacancy centers in diamond: Strengths, limitations, and prospects, J. Appl. Phys. **130**, 150902 (2021).

[25] J. M. Schloss, J. F. Barry, M. J. Turner, and R. L. Walsworth, Simultaneous broadband vector magnetometry using solid-state spins, Phys. Rev. Appl. **10**, 034044 (2018).

[26] L. Rondin, J.-P. Tetienne, T. Hingant, J.-F. Roch, P. Maletinsky, and V. Jacques, Magnetometry with nitrogen-vacancy defects in diamond, Rep. Prog. Phys. **77**, 056503 (2014).

[27] G. J. Abrahams, S. C. Scholten, A. J. Healey, I. O. Robertson, N. Dontschuk, S. Q. Lim, B. C. Johnson, D. A. Simpson, L. C. L. Hollenberg, and J.-P. Tetienne, An integrated widefield probe for practical diamond nitrogen-vacancy microscopy, Appl. Phys. Lett. **119**, 254002 (2021).

[28] M. W. Doherty, N. B. Manson, P. Delaney, F. Jelezko, J. Wrachtrup, and L. C. L. Hollenberg, The nitrogen-vacancy colour centre in diamond, Phys. Rep. **528**, 1 (2013).

[29] F. Jelezko and J. Wrachtrup, Single defect centres in diamond: A review, Phys. Status Solidi (a) **203**, 3207 (2006).

[30] See Supplemental Material at http://link.aps.org/supplemental/10.1103/vjtb-d8kb for additional details on the NV magnetometer calibration procedure and magnetic sensitivity, full experimental dataset, and the ML architecture.

[31] S. F. Ahmed, M. S. B. Alam, M. Hassan, M. R. Rozbu, T. Ishtiak, N. Rafa, M. Mofijur, A. B. M. Shawkat Ali, and A. H. Gandomi, Deep learning modelling techniques: Current progress, applications, advantages, and challenges, Artif. Intell. **56**, 13521 (2023).

[32] Y. LeCun, Y. Bengio, and G. Hinton, Deep learning, Nature (London) **521**, 436–444 (2015).

[33] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, A survey of convolutional neural networks: Analysis, applications, and prospects, IEEE Trans. Neural Networks Learn. Syst. **33**, 6999 (2022).

[34] https://github.com/Fertmeneses/ML_QDM_Meneses_et_al.

[35] K. Sasaki, Y. Monnai, S. Saijo, R. Fujita, H. Watanabe, J. Ishi-Hayase, K. M. Itoh, and E. Abe, Broadband, large-area microwave antenna for optically detected magnetic resonance of nitrogen-vacancy centers in diamond, Rev. Sci. Instrum. **87** (2016).

[36] K. Cabello-Solorzano, I. Ortigosa de Araujo, M. Peña, L. Correia, and J. Tallón-Ballesteros, The impact of data normalization on the accuracy of machine learning algorithms: A comparative analysis, in *International Conference on Soft Computing Models in Industrial and Environmental Applications* (Springer, Berlin, 2023), pp. 344–353.

[37] D. Singh and B. Singh, Investigating the impact of data normalization on classification performance, Appl. Soft Comput. **97**, 105524 (2020).