Positional Health Assessment of Collaborative Robots based on Long Short-Term Memory Auto-Encoder (LSTMAE) Network

Naimul Hasan[1], Louie Webb[1], Malarvizhi Kaniappan Chinanthai[2], Mohammad Al-Amin Hossain[1], Erkan Caner Ozkat[3], Osman M. Tokhi[1], and Bugra Alkan[1]

[1] School of Engineering, London South Bank University, 103 Borough Road, SE1 0AA, London, UK
hasanm56@lsbu.ac.uk, webbl4@lsbu.ac.uk, hossam86@lsbu.ac.uk, tokhim@lsbu.ac.uk, alkanb@lsbu.ac.uk,

[2] Computer Science and Engineering, University of Westminster, 309 Regent Street, W1B 2HW, London, UK
m.kaniappanchinnathai@westminster.ac.uk,

[3] Department of Mechanical Engineering, Faculty of Engineering and Architecture, Recep Tayyip Erdogan University, Türkiye
erkancaner.ozkat@erdogan.edu.tr

Abstract. Calibration is a vital part of ensuring the safety and smooth operation of any industrial robot and this is particularly essential for collaborative robots as any issue pertaining to safety can adversely impact the human operator. Towards this aim, Prognostics and Health Management (PHM) has been widely implemented in the context of collaborative robots to ensure safe and efficient working environments. In this research, as a subset of PHM research, a novel positional health assessment approach based on a Long Short-Term Memory auto-encoder network (LSTMAE) is proposed. An experimental test setup is utilised, wherein the collaborative robot is subject to variations of coordinate system positional error. The operational 3-axis position time-series data of the collaborative robot is collected with the aid of an industrial data acquisition platform utilising influxDB. The experiments show that, with the aid of this approach, manufacturers can assess the positional health of their collaborative robot systems.

Keywords: Collaborative Robotics; Prognostics and Health Management (PHM); Auto-encoder; Wavelength Scattering; LSTM; Machine Learning; Manufacturing Assembly.

1 Introduction

The term ‘Collaborative Robot’ (Cobot) refers to a robot that can safely do tasks in close proximity to humans [1]. They have a wide range of applications across multiple sectors such as product assembly, product packaging, material handling, welding, material removal, defect and quality inspections. In order to integrate them in advanced manufacturing systems to work alongside humans, it is essential that they are regularly calibrated and serviced to prevent health and safety issues [2]. Particularly, issues arising from positional encoders, defects in the robot-base securing system, or propagation of abnormal vibrations could manifest as functional failure of the cobot that might not only lead to a drop in quality and production efficiency, but also compromise the safety of human operators [3]. Therefore, it is critical to have a solid understanding of the positional degradations of a cobot such that anomalous trajectories can be flagged before they have the chance to propagate into a serious fault.

To address the above-mentioned problem, Prognostics and Health Management (PHM) can be applied in cobots to support maintenance decision-making. They can help in reliable monitoring, detection of incipient defects, and forecasting of future faults. PHM can be implemented at two distinct levels: *i*) the component level, in which it is typically used to monitor the health of specific components (such as gears, engines, and electronic devices); and *ii*) the system level, in which it is employed to assess the health of the overall system (such as robots, and workstations) by taking into consideration a variety of aspects, including system responses and process-related metrics [4]. The following paragraphs provide a brief review of the relevant literature and highlight the research gap that this article aims to fulfill.

Wang et al., [5] proposed a deep learning (DL) architecture based on the vibration signal of rolling elements, which is de-noised using the combined use of self-attention (SA) mechanisms and a bidirectional long- and short-term memory (BiLSTM) network. A similar approach was presented in [6] in which a CNN-LSTM deep neural network called MC-LSTM architecture was used instead of SA to detect collisions (collision points) from cobots relying on rotor channel estimation statistics. Furthermore, Nabissi et al., [7] proposed a solution-based Robot Operating System (ROS) for automatic failure detection and conditional monitoring (CM) of cobots. Their architecture can detect joint anomalies using torque information to define standard health indicators (HIs), depending on whether the condition is highly dynamic or not. Auto-encoders are also utilised extensively in the PHM of industrial robots and cobots. Polenghi et al., [3] proposed a hybrid DL-based architecture for fault detection of cobots following PHM guidelines; a clustering algorithm was used on three different trajectories followed by the use of an autoencoder to identify unhealthy trajectories. A final clustering algorithm was used for functional failure trajectories. Similarly, in paper [4], by Qiao et al., followed PHM guidelines to identify the positional health changes of industrial robots. However, instead of the traditional DL mechanism, advanced sensors, visualization tools, and algorithms were used. On the other hand, Yun et al., [8] proposed an autoencoder-based architecture for inconsistency detection using the sound sensor in robot limbs. Sound sensors were implemented as a solution to address the problem of a limited quantity of defect data. A method that converts two-dimensional sound signals into one-dimensional sound signals using the short-term Fourier transform (STFT). The STFT signals that were considered as normal were sent to the training autoencoder; STFT signals that were both normal and anomalous were sent to the feed-forward part of the training encoder so that it could determine the difference between the reconstructed features. In another experimental study, the use of an autoencoder for the purpose of anomaly detection in cobots was conducted by Graabaek et al. [9]. They compared various outlier detection methods for cobot pick-and-place operations, including k-Nearest Neighbors (kNN), Local Outlier Factor (LOF), Randomized Neural Network (RandNet), Long Short-Term Memory (LSTM), and Bilinear (combination of CNN and LSTM).

The vast majority of the approaches that are currently available are utilised for the purpose of managing component-level fault detection of cobots such as drivers, controllers, sensors, etc., [3,4]; however, only a select few of these methods are capable of being integrated at the system level [3]. There is a lack of research in the field of positional health assessment for cobots as the majority of the published works in the literature do not concentrate on the concept of predictive maintenance for cobot systems but instead focus on collision detection and trajectory planning [3]. As positional health anomalies are inherently not classified as faults by the cobot, it is much more difficult to detect them than other types of system anomalies. However, identifying them can assist in avoiding collisions, errors in assembly tolerance, and product damage [4].

Therefore, this paper proposes a novel method for detecting anomalies from end-effector position data of cobots using an LSTM-auto-encoder network. The proposed method is illustrated through the use of a case study that features a simple pick-and-place operation that incorporates two variations of a positional fault. The findings demonstrated that the proposed method is capable of detecting anomalies from the 3-axis positional trajectory such that associated safety risks and production quality issues can be avoided. The remaining sections of the paper are structured as follows: i) the proposed auto-encoder-based solution is described in greater depth in Section 2, ii) the testing environment is outlined in Section 3, iii) a discussion of the approach's performance and experimental results are presented in Section 4, and iv) the final section wraps up the paper and lays out the plan for further research.

2 Research Methodology

2.1 Long Short-Term Memory (LSTM) Autoencoder

An autoencoder is a type of neural network that is trained to reconstruct its input data. The idea is to train the network to encode the input data into a lower-dimensional representation and then decode this representation back into the original data. By doing so, the network can learn to capture the most important features of the input data, which can be useful for tasks such as data compression or anomaly detection. LSTM is a type of recurrent neural network (RNN) that is designed to handle the vanishing gradient problem in traditional RNNs [10]. The vanishing gradient problem occurs when the gradients used to update the weights in the network become very small, making it difficult for the network to learn long-term dependencies [11]. LSTMs solve this problem by introducing a set of memory cells that can store information over long periods of time, allowing the network to learn long-term dependencies more effectively [12].

Combining LSTM and autoencoder results in an LSTM autoencoder (LSTMAE), which can be used for tasks such as sequence-to-sequence prediction, anomaly detection or feature extraction. The LSTM encoder takes in a sequence of inputs and encodes them into a lower-dimensional representation using the LSTM memory cells. The LSTM decoder then takes this representation and decodes it back into the original sequence. During training, the network is trained to minimize the difference between the original sequence and the reconstructed sequence.

2.2 The proposed architecture

Figure 1 shows the proposed LSTMAE architecture. The proposed network involves a two-stage process that transforms data into a matrix of frames coupled with LSTM autoencoder. The first stage involves organising the input data into overlapping frames. The second stage involves feeding the organised data into an LSTM autoencoder to reconstruct the original signal. **Figure 2** shows the flowchart of the proposed approach.

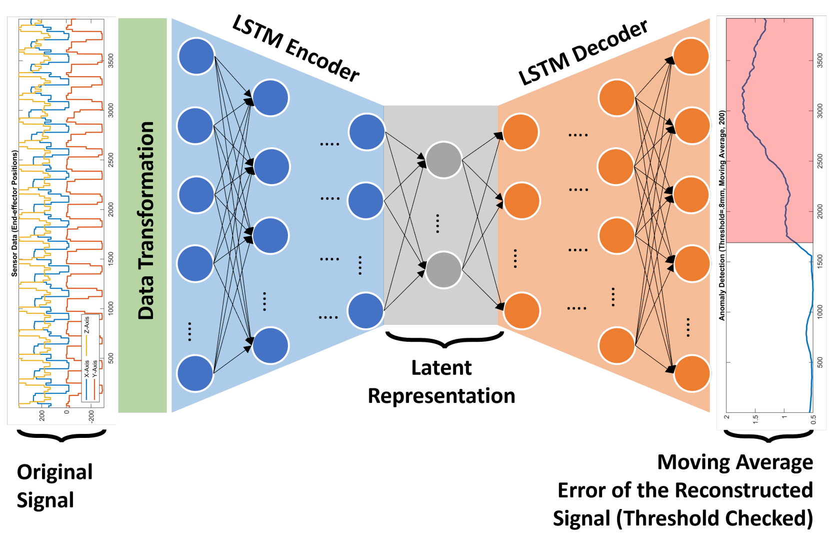

Fig. 1: The proposed LSTMAE model.

In stage one, overlapping frames for the input signal as part of the data transformation are first created. Following that, the dataset is split into training and testing data, where the latter consists of both normal and abnormal signals.

The z-score normalisation, as shown in Equation 1, was used on the training set and provided as input to the LSTMAE.

$$z = (x - \mu)/\sigma \tag{1}$$

where z is the normalized signal, x is the input signal, μ and σ are the mean and the standard deviation of the input signal, respectively.

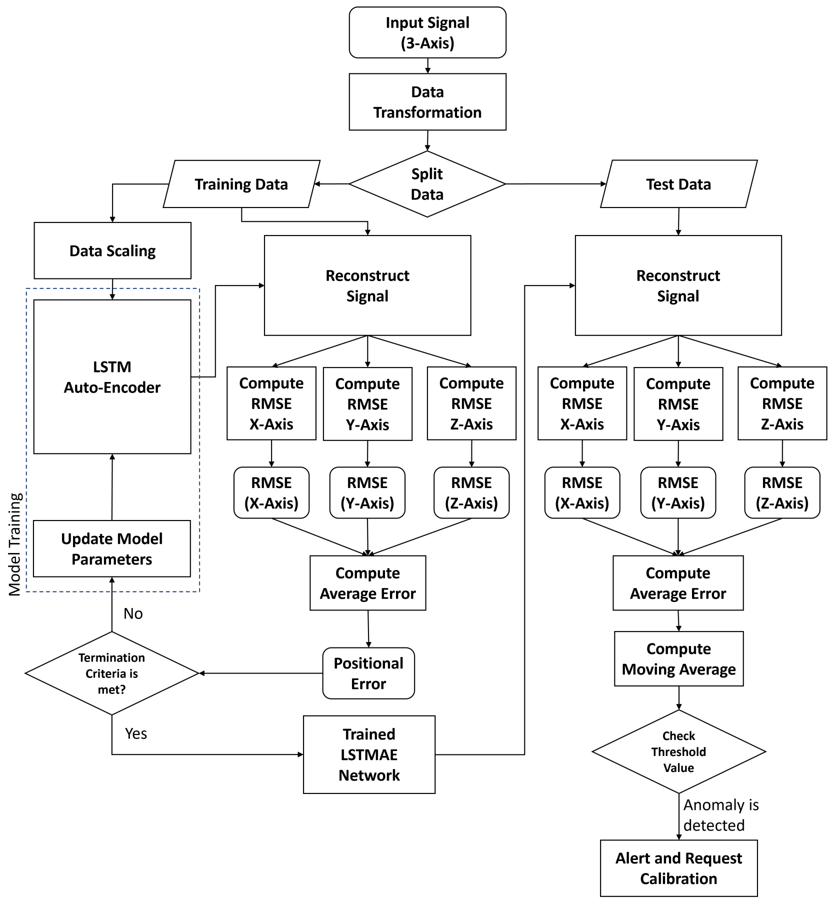

Fig. 2: The flow-chart of the proposed LSTMAE model.

In stage 2, an autoencoder that can learn to reconstruct the input data with high accuracy while at the same time capturing complex patterns and dependencies from the bottleneck is employed. The network's encoding phase comprises a series of interconnected layers, starting with an LSTM layer, followed by a ReLU layer, a second LSTM layer, a dropout layer and a final ReLU layer. In order to address the issue of diminishing gradients, the integration of ReLU layers has been implemented to facilitate non-linear transformations. Additionally, a dropout layer has been utilised during training to prevent overfitting by randomly discarding 20% of the input. To achieve the reconstruction of the input sequence, a custom layer was incorporated into the decoding stage of the network. This layer was specifically designed to emulate the output of the final LSTM layer in the encoding stage. The architecture of the decoding stage is composed of an initial Long Short-Term Memory (LSTM) layer, succeeded by a dropout layer, a Rectified Linear Unit (ReLU) layer, a second LSTM layer, and a final ReLU layer. In order to produce the predicted values, the regression layer receives input from the ultimate decoder layer. It is noteworthy that the arrangement of LSTM cell numbers in the decoder stage is reversed in comparison to that of the encoder stage.

Once the autoencoder reconstructs the signal, the Root Mean Square Error (RMSE) metrics for each axis is calculated. The average of the three RMSE values is then calculated as the 'average prediction error'. The RMSE calculation is shown in Equation 2 as

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{x_i} - x_i)^2} \tag{2}$$

where N is the total number of input signal, $\hat{x_i}$ and x_i are the predicted and input signal, respectively. Hyperparameter tuning of the LSTMAE parameters such as the optimiser and activation function using grid search was performed.

A moving average technique is applied on the 'average prediction error' for timeseries smoothing. From this processed timeseries data, an anomaly can be detected if the prediction error exceeds a threshold value. Once the anomaly is detected, appropriate measures to deal with the positional fault such as alerting the human operator and requesting calibration can be carried out.

3 Experiments

3.1 Experimental test setup

The experimental test setup consists of a small-scale cobot and a laptop PC that collects the real-time position of the end-effector within the coordinate system. The computer's Intel(R) Core(TM) i7-11850H CPU @ 2.60 GHz and 32 GB RAM were used for all operations in the MATLAB environment. The cobot is equipped with a vision system and adaptive gripper to help pick objects of different shapes and colours. **Figure 3** shows the test setup where the robot picks the workpieces and places them in a container.

3.2 Process Description

The considered pick and place operation comprises of the human operator first placing workpieces within the cobot's workspace. The cobot identifies and locates

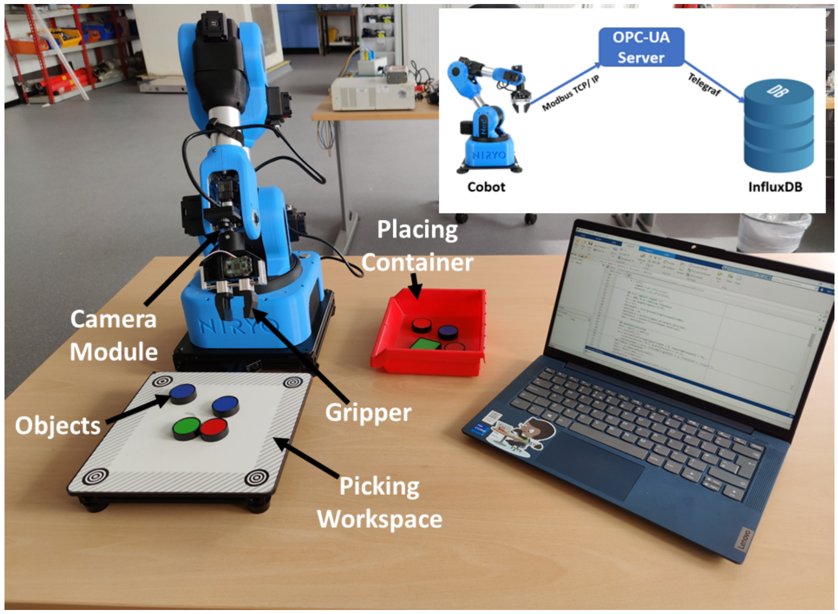

Fig. 3: Experimental test setup.

the workpieces by their shape, colour, and position in relation to the calibrated workspace coordinates with the help of a vision system. The identified workpiece is then picked and placed in a container. These steps are repeated for 20 cycles, where the first 10 cycles are working within normal calibrated conditions. Two variations of the fault are introduced at the 11th and 15th cycles, respectively, to create an offset in the cobot's workspace coordinates. Due to this reason, the cobot's end-effector position will have an offset but the cobot does not realise this and still continues working according to its process plan.

3.3 Faults

The cobot picks and places the object depending on the coordinates of the workspace. Hence, any offset in the cobot's coordinate system make it difficult or impossible in some cases for the gripper to pick the object. This can be due to functional errors or hardware-related errors such as encoder faults. In the experiment, two variations of coordinate positional error are introduced by making changes to the cobot's software to mimic encoder faults. It is important to note that the actual position of the cobot, the container and the workspace are not modified in any way. The first error is a 1mm offset in the x,y and z axes that was introduced at time unit 1970, and the second error is introduced at time unit 2924 with 5 mm offset in the x,y and z axes.

3.4 Data Acquisition

On the top right corner of **Figure 3**, the diagram displays the information flow and acquisition technologies used by the cobot to create a time series database. The robot can communicate with the OPC-UA (Open Platform Communications—Unified Architecture) server via Modbus, where the real-time positional measurements can be read within the OPC-UA-Modbus-TCP client-server. This allows OPC-UA to map each reading with unique identifiers within the same namespace, further allowing protocol conversion and data access controls thus providing a standardised and interoperable interface between the robot and time series database. When storing the reading, the combination of Telegraf and InfluxDB was chosen. InfluxDB is an open-source, high-performance, and scalable time-series database designed for handling large volumes of time-stamped data. It is specifically optimised for storing, querying, and visualising time-series data, which typically consists of data points associated with timestamps, such as sensor readings, metrics, logs, events, and other time-stamped data generated by various applications and systems. Telegraf is a data collection agent developed by InfluxData that is designed to gather, process, and send data to InfluxDB. It acts as a bridge between various data sources and InfluxDB, further enabling seamless data ingestion into InfluxDB for storage, querying, and visualisation. Therefore, within the experiment, positional measurements are identified as separate measurements and stored within the same database at a sampling rate of 50 ms.

3.5 Implementation of methodology

In the experiment, the end-effector position (i.e., gripper) of the cobot was collected as a form of raw time series signal data for three axes (X, Y, and Z). A training dataset was used that covered the period of time from time unit 0 to 1969 timeunits, and the remaining data was used to create a test dataset. Each timeunit represents 100 milliseconds. For stage one of the methodology, in order to perform data transformation, frame length of 5 timeunits is taken. Following the data transformation, the timeseries is input to stage 2 where the LSTM autoencoder is be used to reconstruct the signal. The L2WeightRegularization, SparcityRegularization, and SparsityProportion were selected as 1.0e-10, 1.0e-10, and 0.7, respectively. The model converged at 2000 epochs and the output reconstructed signal is explained in more detail in the next section.

4 Results

Figure 4 shows a timeseries profile of the end-effector position on the X axis at the top. The bottom profile in the figure, represents the calculated RMSE error in the X-axis. The higher the error, the more the end-effector position deviates from the normal. The variations in the X axis until 1900 are the result of the minor deviations attributed to the presence of a human operator interacting with the cobot's workspace. However, from approximately 1970 timeunits, the variants are more prominent. Again at 2200 timeunits, there is an increase in the variation. This can be attributed to the introduction of the functional error as an offset in the cobot's workspace. Similarly, from **Figure 5**, the increase in the RMSE error can be seen from 1900 and then at 2200. From **Figure 6**, a similar pattern emerges. Following this, the average of the three RMSE values is taken as the 'average prediction error' which is shown in (**Figure 7**); it can be seen that the 'average prediction error' increases after the introduction of the positional faults at 1970. A threshold of 0,6 is set for the detection of anomaly and at 2222 timeunits, an alert will be generated to indicate that there is some issue with the end-effector position. Although such faults might not normally be captured by the cobot, the use of such a methodology can help continuous monitoring of the cobots's end-effector positions. Therefore, when very small deviations, such as 1mm, is introduced into the workspace coordinates, the proposed approach is capable of detecting such anomalies using timeseries data and generating alerts before they can progress into more serious issues.

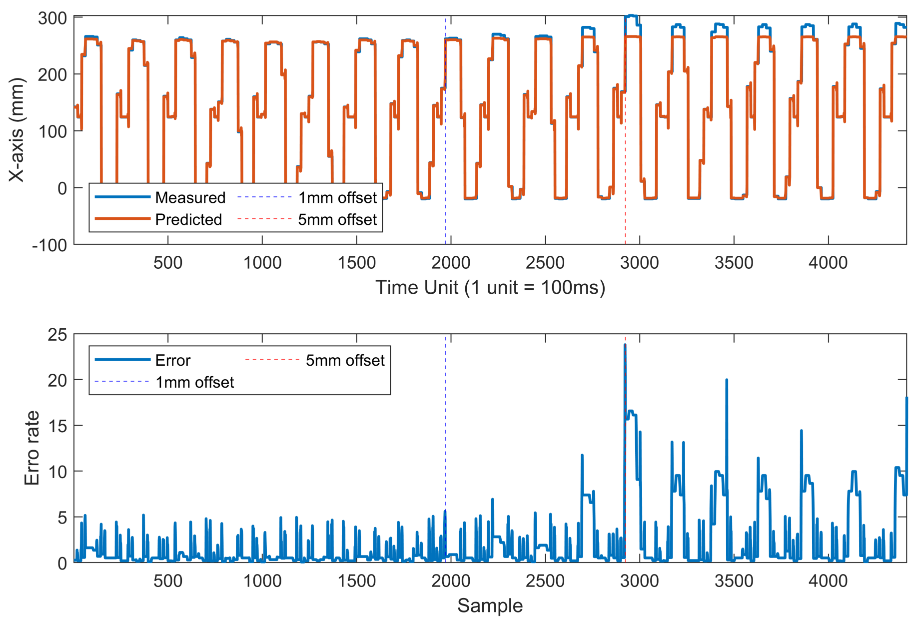

Fig. 4: Error Prediction in X-axis

4.1 Discussions and Limitations

This section presents a brief evaluation of the proposed methodology and it is important to highlight that the key focus of the LSTMAE for positional health

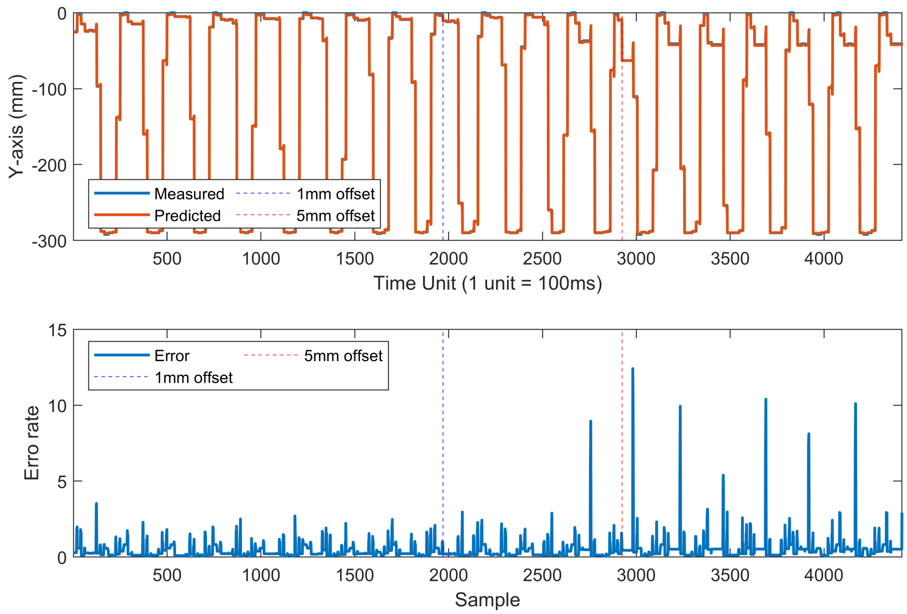

Fig. 5: Error Prediction in Y-axis

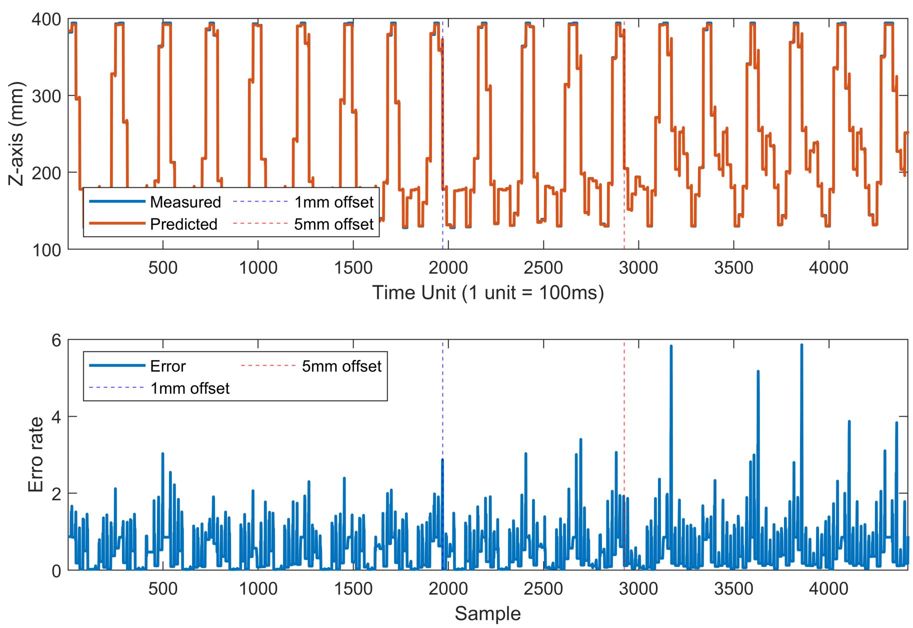

Fig. 6: Error Prediction in Z-axis

assessment is to identify faults before it manifests itself into more serious problems in the industrial system. From the above-mentioned results, it can be seen

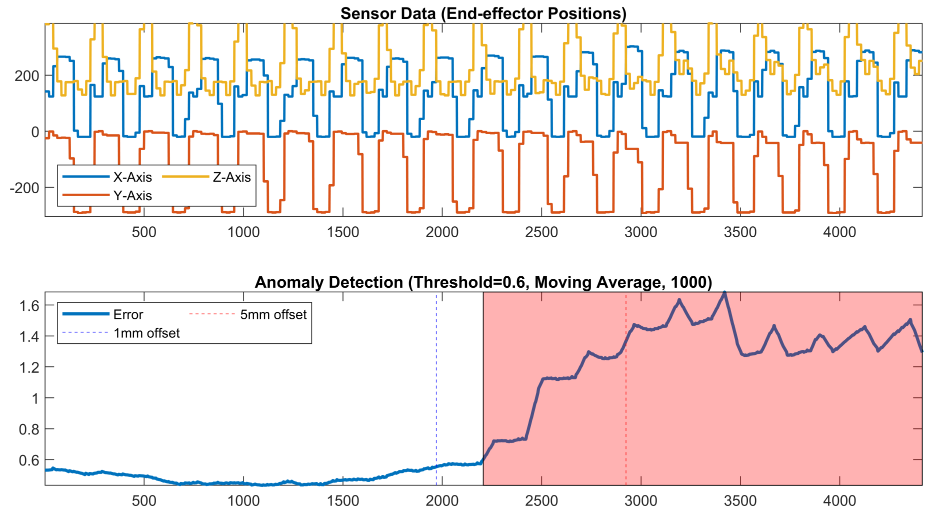

Fig. 7: Anomaly detection results.

that sudden changes in the timeseries profile can be detected as anomalies using the end-effector positional measurements. However, there is a small lag, approximately 200ms, Although this needs to be researched further using different use cases, it is a promising area to improve the quality and safety of human-robot collaboration.

The authors would like to highlight a few limitations of the proposed work. The time series data collection with the cobot depends on the process we set up for it to perform. In our experiment, we used a simple pick and place operation which generates a dataset with limited stochasticity. Therefore, the raw data was used as input to the LSTMAE architecture. However, if the initial dataset has a lot of variations, it might not be possible to use the dataset as such; more complex techniques such as wavelet scattering might need to be employed before that dataset can be input into the LSTMAE architecture. The proposed methodology is limited in its scope to positional health assessment of cobots and only sudden failures are considered. Propagating errors that can be interpreted from timeseries are not considered in this work.

5 Conclusions

This research presents the novel application of an LSTMAE network in the positional health assessment of cobots. The approach comprises of two stages wherein the positional measurement values of the cobot end-effector are subject to data transformation and used to train the LSTMAE network. During the training process, RMSE is used as the performance measure to evaluate the network. The trained network is then used to predict faults by detecting anomalies which can then be used to alert and request calibration. The core contribution of the research is pertaining to positional anomalies which is a difficult failure to detect. However, timely detection of such failures helps prevent health & safety hazards and ensures product quality. Typically, cobots do not follow strict state transitions/processes as they need to accommodate working with a human operator. This means that they are subject to process deviations and hence the research can be extended by using techniques such as process mining to represent the factual robotic process and state transitions. Another future work of the research is the determination of the threshold at which the timeseries values can be considered as an anomaly. Furthermore, the authors believe that the performance of the LSTMAE network can be improved by fine-tuning the process parameters using different hyperparameter tuning approaches.

References

1. Peshkin, M., Colgate, J., Wannasuphoprasit, W., Moore, C., Gillespie, R. & Akella, P. Cobot architecture. *IEEE Transactions On Robotics And Automation.* **17**, 377-390 (2001)
2. Djuric, A., Urbanic, R. & Rickli, J. A framework for collaborative robot (CoBot) integration in advanced manufacturing systems. *SAE International Journal Of Materials And Manufacturing.* **9**, 457-464 (2016)
3. Polenghi, A., Cattaneo, L. & Macchi, M. A framework for fault detection and diagnostics of articulated collaborative robots based on hybrid series modelling of Artificial Intelligence algorithms. *Journal Of Intelligent Manufacturing.* pp. 1-19 (2023)
4. Qiao, G., Schlenoff, C. & Weiss, B. Quick positional health assessment for industrial robot prognostics and health management (PHM). *2017 IEEE International Conference On Robotics And Automation (ICRA).* pp. 1815-1820 (2017)
5. Wang, Y., Cao, G. & Han, J. A Combination of Dilated Self-Attention Capsule Networks and Bidirectional Long-and Short-Term Memory Networks for Vibration Signal Denoising. *Machines.* **10**, 840 (2022)
6. Czubenko, M. & Kowalczuk, Z. A simple neural network for collision detection of collaborative robots. *Sensors.* **21**, 4235 (2021)
7. Nabissi, G., Longhi, S. & Bonci, A. ROS-Based Condition Monitoring Architecture Enabling Automatic Faults Detection in Industrial Collaborative Robots. *Applied Sciences.* **13**, 143 (2023)
8. Yun, H., Kim, H., Jeong, Y. & Jun, M. Autoencoder-based anomaly detection of industrial robot arm using stethoscope based internal sound sensor. *Journal Of Intelligent Manufacturing.* **34**, 1427-1444 (2023)
9. Graabæk, S., Ancker, E., Christensen, A. & Fugl, A. An Experimental Comparison of Anomaly Detection Methods for Collaborative Robot Manipulators. (TechRxiv,2022)
10. Pascanu, R., Mikolov, T. & Bengio, Y. On the difficulty of training recurrent neural networks. *International Conference On Machine Learning.* pp. 1310-1318 (2013)
11. Ribeiro, A., Tiels, K., Aguirre, L. & Schön, T. Beyond exploding and vanishing gradients: analysing RNN training using attractors and smoothness. *International Conference On Artificial Intelligence And Statistics.* pp. 2370-2380 (2020)
12. Landi, F., Baraldi, L., Cornia, M. & Cucchiara, R. Working memory connections for LSTM. *Neural Networks.* **144** pp. 334-341 (2021)