A Digital Twin-Driven Ultra-Precision Machining System for Future Smart Manufacturing

By

Charles Walker

Supervisor - Xichun Luo

A thesis submitted to the University of Strathclyde

for the degree of

Doctor of Philosophy

Centre for Precision Manufacturing

Department of Design, Manufacturing and Engineering Management

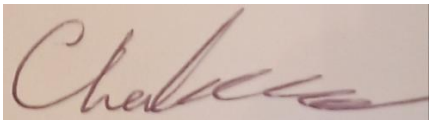
Faculty of Engineering

University of Strathclyde

Jan 2026

Declaration Statement

This thesis is the result of the author's original research. It has been composed by the author and has not been previously submitted for examination, which has led to the award of a degree.

Signed:  Date: 31/01/2026

List of Publications

Journal papers:

[1] Abhilash Puthanveettil Madathil, Luo, X., Liu, Q., Rajeshkumar Madarkar and **Walker, C**. (2024a). A Fingerprint Approach for Computationally Efficient Digital Twins of Precision Manufacturing Processes. *MATEC Web of Conferences*, 401, pp.13004–13004. doi:https://doi.org/10.1051/matecconf/202440113004.

[2] Abhilash Puthanveettil Madathil, Luo, X., Liu, Q., **Walker, C**., Rajeshkumar Madarkar, Cai, Y., Liu, Z., Chang, W. and Qin, Y. (2024b). Intrinsic and post-hoc XAI approaches for fingerprint identification and response prediction in smart manufacturing processes. *Journal of Intelligent Manufacturing*, 35, pp.4159–4180. doi:https://doi.org/10.1007/s10845-023-02266-2.

[3] Abhilash Puthanveettil Madathil, **Walker, C**., Luo, X., Liu, Q., Rajeshkumar Madarkar and Qin, Y. (2024c). XAI-driven digital twin for cobot dynamic error compensation. *Procedia CIRP*, 126, pp.176–181. doi:https://doi.org/10.1016/j.procir.2024.08.320.

[4] Chen, X., Luo, X., Sun, Y., Zhong, W., **Walker, C**., Tian, Y., Wang, Z., Chang, W. and Wardle, F. (2026). A novel bio-inspired compound restrictor for high-precision aerostatic bearings: design and evaluation. *Precision Engineering*, 97, pp.348–366. doi:https://doi.org/10.1016/j.precisioneng.2025.09.025.

[5] Madarkar, R., Luo, X., **Walker, C**., Puthanveettil Madathil, A. and Liu, Q. (2024). Prospects of Digital Twin for Dynamic Life Cycle Assessment of Smart Manufacturing Systems. *MATEC Web of Conferences*, [online] 401, p.13006. doi:https://doi.org/10.1051/matecconf/202440113006.

[6] Puthanveettil Madathil Abhilash, Luo, X., Liu, Q., Rajeshkumar Madarkar and **Walker, C**. (2024). Towards next-gen smart manufacturing systems: the explainability revolution. *Deleted Journal*, [online] 1(1). doi:https://doi.org/10.1038/s44334-024-00006-9.

[7] Puthanveettil Madathil, A., Luo, X., Liu, Q., **Walker, C**., Madarkar, R. and Qin, Y. (2025). A review of explainable artificial intelligence in smart manufacturing. *International Journal of Production Research*, 63, pp.1–44. doi:https://doi.org/10.1080/00207543.2025.2513574.

Conference papers:

[1] **Walker, C**., Digital Twin for CNC Machining: A Review. The 41st MATADOR Conference and AMPI Launch, Manchester, UK, 15-17 Sep 2021.

[2] **Walker, C** Luo, X., Kundu, P., Chang, W., Xie, W., Ball, P., Badakhshan, E, Digital Twin Technology for CNC Machining: A Review," *2022* 8th International Conference on Nanomanufacturing & 4th AET Symposium on ACSM and Digital Manufacturing (Nanoman-AETS), Dublin, Ireland, 30 Aug- 1 Sep 2022, pp. 1-6, doi: 10.1109/Nanoman-AETS56035.2022.10119512.

[3] **Walker, C**, Luo, X., Abhilash P M, Liu, Q., Madarkar, R., Yang, E., Digital twin of dynamic error of a collaborative robot, in euspen's 23rd International Conference & Exhibition, Copenhagen, Denmark, 12-16 June 2023. (**Heidenhain Scholarship**).

[4] **Walker, C**. Puthanveettil Madathil, A., Luo, X**.**, Madarkar, R., Liu, Q., A digital twin-driven ultra-precision Machining System, 24th International Conference of Manufacturing Research, Glasgow, UK, 28-30 Aug 2024, In: *MATEC Web of Conferences*, 401, p. 13009. doi:10.1051/matecconf/202440113009.

[5] **Walker, C**., Digital Twin-Driven Work Handling, The 20thInternational Conference on Precision Engineering, Sendia, Japan, 23-26 Oct 2024.

Abstract

The UK manufacturing industry is undergoing digital transformation to improve productivity and support the production of high-value components required for sectors such as energy, transport, and healthcare. In ultra-precision manufacturing, productivity is limited by dynamic errors caused by machine motion, structural compliance, and inertial effects. Many of these errors occur outside the machine servo control loop and cannot be measured or corrected using conventional encoder-based feedback systems, limiting achievable machining accuracy, automation, and production speed.

This research investigates a digital twin-driven approach for measuring, predicting, and compensating for dynamic errors in ultra-precision manufacturing systems. The proposed framework integrates a collaborative robot (COBOT) and a CNC hybrid milling machine, enabling automated workpiece handling and machining within a coordinated digital twin architecture.

A novel sensing methodology using triaxial accelerometers is developed to measure machine dynamic behaviour outside the control loop. Machine displacement is estimated through signal processing techniques, including fast Fourier transform (FFT) filtering and double integration of acceleration data. Experimental validation using ball bar testing demonstrates that this approach improves COBOT tracking accuracy, achieving an average improvement of 41.3% compared with encoder-based measurements.

To enable real-time implementation, explainable artificial intelligence (XAI) models are introduced to predict dynamic error directly from acceleration signal features. Using the QLattice modelling approach, interpretable mathematical expressions are generated to estimate dynamic errors with reduced computational requirements. Experimental evaluation shows that the COBOT digital twin reduces placement error in automated pick-and-place tasks by 31.5%

for MoveJ motions and 4% for MoveL motions. The Move Joint (MoveJ) commands, where the robot follows the most efficient joint path between two points, and Move Linear (MoveL) commands, where the Tool Centre Point (TCP) moves in a straight line between two positions.

Application of the approach to the hybrid mill demonstrated the ability to detect machine dynamic behaviour; however, reliable dynamic error correction at micron-level precision was limited by sensor resolution, accelerometer frequency response, and environmental variation.

A proof-of-concept smart manufacturing demonstrator integrating the COBOT and hybrid mill was implemented using a CubeSat end cap component. While full closed-loop corrective machining was not achieved, the system successfully demonstrated coordinated robotic handling and machining within a digital twin framework.

Overall, this research demonstrates that accelerometer-based digital twins combined with explainable AI provide a viable approach for measuring and compensating dynamic errors outside the control loop, particularly for robotic workpiece handling. The work establishes a foundation for future development of predictive digital twin systems capable of improving precision, automation, and productivity in ultra-precision manufacturing.

Acknowledgement

I would like to take this opportunity to thank my supervisor, Professor Xichun Luo, for the continued support, guidance, and opportunities he has provided throughout my PhD. His mentorship has enabled me to develop my engineering and research capabilities to a new level and has also allowed me to attend international events and compete in international competitions on behalf of the University of Strathclyde. I would also like to thank my secondary supervisor, Dr Erfu Yang, for his support during my time at Strathclyde and for recommending me for the PhD programme.

I am grateful to the University of Strathclyde for awarding me the John Anderson Research Award (JARA) Studentship, which supported my PhD studies.

I would like to extend my sincere appreciation to Dr Abhilash Puthanveettil Madathil, Dr Rajesh Madarkar, and Dr Qi Liu for their advice and support throughout my PhD, as well as to all members of the Centre for Precision Manufacturing at the University of Strathclyde. I am also deeply grateful to the technical support staff at the University of Strathclyde for their assistance with my research and for their support during competitions and related projects.

Finally, I would like to thank my partner, family, and friends for their encouragement and support over the years, and for helping to make the process of completing a PhD as manageable as possible.

The authors would also like to thank the Engineering and Physical Sciences Research Council (EPSRC) (EP/K018345/1, EP/T024844/1, EP/V055208/1, W004860/1) for providing financial support for this research.

Table of Contents

Table of Figures

Table of Tables

Abbreviations

ABC — Artificial Bee Colony

ACC — Accelerometer

AI — Artificial Intelligence

ANN — Artificial Neural Networks

CCW — Counter-Clockwise

COBOT — Collaborative Robot

CMM — Coordinate Measuring Machine (CMM)

CNC — Computer Numerical Control

CSV — Comma-Separated Values

CW — Clockwise

DT — Digital Twin

DTree — Decision Trees

FAE — Finite Element Analysis

FFT — Fast Fourier Transform

FNN — Fuzzy Neural Network

GA — Genetic Algorithm

IoT — Internet of Things

KNN — K-Nearest Neighbours

KPI — Key Performance Indicators

LISA — Line Information System Architecture

MAE — Mean Absolute Error

MAV — Mean Absolute Value

ML — Machine Learning

NC — No Compensation

NURBS — Non-Uniform Rational B-Spline

PCB — Printed Circuit Board

PID — Proportional–Integral–Derivative

PLA — Polylactic Acid

PLC — Programmable Logic Controller

PSO — Particle Swarm Optimisation

RC — Radial Corrections

RFR — Random Forest Regression

RMS — Root Mean Square

RMSE — Root Mean Square Error

RUL — Remaining Useful Life

TCN-BiLSTM — Temporal Convolutional Network and Bidirectional Long Short-Term Memory

TCP — Tool Centre Point

WC — With Compensation

XAI — Explainable Artificial Intelligence

XGB — Extreme Gradient Boosting

Chapter 1 – Introduction

1.1 Research background

The UK manufacturing industry stands on the threshold of a technological revolution and digital transformation, which are considered key drivers for improving productivity [1], particularly for high-value-added products required to address some of the greatest global challenges, such as energy, the environment, transportation, and an ageing population. Ultra-precision manufacturing can improve productivity in the manufacture of turbine blades for more efficient energy production, enable the machining of superalloys and ceramics for aerospace applications, and achieve greater precision in medical implants such as knee replacements. The design, manufacture, assembly, and characterisation challenges associated with these products are considerable, requiring a step change in the current manufacturing systems to achieve the ambitious target of securing industrial efficiency gains of up to 25% [2]. Britain's productivity has long lagged behind that of its competitors. Digital twin is regarded as one of the key digital manufacturing technologies enabling productivity gain for the UK manufacturing industry.

A digital twin is a digital representation of a manufacturing asset, system, process, or environment that allows the inclusion of a two-way communication flow into and out of the real world in a timeframe that is appropriate for the required decisions [3].

Currently, digital twins have been applied in manufacturing to improve the effectiveness of CNC machining through a range of measures, including prediction of tool wear, optimisation of machining conditions, and compensation of trajectory errors. Examples include digital twin frameworks for tool wear prediction using knowledge embedding [4], optimisation of feed rate and spindle speed through simulation-based digital twins [5], and the prediction and compensation of trajectory errors using artificial intelligence and historical machining data [6].

These approaches exploit the real-time capabilities of digital twins to monitor, track, and correct CNC machine behaviour. However, they are predominantly focused on conventional machining processes and have not been widely extended to the more demanding domain of ultra-precision manufacturing. In ultra-precision machining, tolerances are significantly smaller, and system dynamics are more complex, making conventional digital twin implementations insufficient. Consequently, there is a clear need for advanced digital twin solutions capable of improving both production rate and part quality in ultra-precision manufacturing through the real-time reduction of dynamic machining errors.

In CNC machining, dynamic error is the deviation between the tool and the reference (setpoint) displacements. It includes both servo response (following) errors and mechanical deviations associated with the dynamic response of mechanical structures and interfaces (i.e., dynamic errors inside and outside the servo loop). The dynamic error outside the servo loop cannot be measured by the machine encoder.

This thesis will develop and implement a digital twin-driven dynamic error control approach to predict and correct dynamic errors in an ultra-precision machining system, including a COBOT used for work handling and a CNC hybrid mill. The research will contribute to the ultimate goal of establishing a smart manufacturing system for ultra-precision machining operations in the future.

1.2 Aim and objectives

This research aims to investigate the feasibility of a digital twin–driven control framework for ultra-precision machining systems that enables real-time detection and correction of dynamic positioning errors during machining and automated workpiece handling.

The principal objectives are:

1. To gain an in-depth understanding of state-of-the-art research on digital twins for advanced manufacturing and to identify knowledge gaps of digital twins for ultra-precision manufacturing.
2. To identify an effective measurement approach to obtain machine dynamic errors that are not measurable by machine encoders for CNC machining and workpiece handling.
3. To create a digital twin-driven dynamic error compensation approach for work handling and machining operations for an inline application.
4. To validate the effectiveness of the proposed digital twin-driven dynamic error compensation approach on an ultra-precision machining system through practical testing.

1.3 Structure of thesis

Chapter 2 presents a state-of-the-art literature review on the development of digital twins for manufacturing, machining errors and the specific challenges associated with ultra-precision manufacturing.

Chapter 3 outlines the research methodology for establishing a digital twin-driven ultra-precision machining system.

Chapter 4 presents the dynamic error tracking test used to measure the displacement at the tool centre point (TCP) of the COBOT, validating the suitability of this approach for improving motion tracking in machines with differing control architectures.

Chapter 5 presents the dynamic error tracking test used to measure the displacement at the TCP of the hybrid mill, validating the suitability of this approach for improving motion tracking in machines with differing control architectures.

Chapter 6 presents the predictive digital twin-driven dynamic error control for the COBOT and the hybrid mill using machine learning, covering the training and validation testing of the error reduction.

Chapter 7 describes the experimental testing of the predictive digital twin-driven dynamic error compensation approach on a COBOT for pick-and-place operations. The effectiveness of the proposed approach is evaluated during varied tasks.

Chapter 8 presents the demonstration of the proposed predictive digital twin-driven dynamic error compensation approach through a showcase of machining cube satellites on an established digital twin-driven ultra-precision machining system.

Chapter 9 concludes the thesis by summarising the findings, evaluating the overall contributions of this work, and outlining directions for future research in the development and application of digital twins for ultra-precision manufacturing.

2 Chapter 2 – Literature review

2.1 Introduction

Emerging digital twin technology, along with big data, the Internet of Things (IoT), artificial intelligence (AI), and cloud computing, is a key enabler for realising the digital transformation of our society, ultimately leading to an improved quality of life. The digital twin paves the way for cyber-physical integration and serves as a bridge between the physical and cyber worlds. It provides manufacturing enterprises with a new approach to smart production and precision management[7]. As interest and research in digital twins for manufacturing have increased, there is a growing need for clearly defined categories. This will facilitate a better understanding of the capabilities and functions of digital twins, simplify development and promote wider adoption.

The term "digital twin" was first introduced by Michael Grieves in 2003 within the context of product lifecycle management. This concept was further elaborated in a white paper published in 2014 [8], where a digital twin is described as consisting of three main components: the physical model in real space, the virtual model in virtual space, and the data and information flow that connects the two.

Digital twins are used for greater data collection and processing, so a physical system can be controlled better through a virtual system. This allows for simulations to be carried out to monitor and predict the physical system. These simulations can help provide information on maintenance requirements, efficiency, productivity, and real-time status updates. Due to this ability to increase the efficiency of a system, the application of digital twins will be universal to monitor any system or product throughout its life span [9, 10]. So far, applications of digital twin are focused on the manufacturing system level [11] and deal with production planning and control, monitoring, assessment management, quality and maintenance[12-24]. In recent

years digital twin technologies have been further developed for machine tool condition prediction[25], tool wear monitoring[26, 27], modelling machine tools[28, 29] and real-time simulation of the CNC machining process for quality control[16, 30], cloud manufacturing for computational and network demands [30, 31], safety in collaborative workspaces [32], for controlling and monitoring an automated welding arm [33], autonomous monitoring and troubleshooting [19], prediction of performance degradation [34], additive manufacturing [35].

This chapter will proceed as follows: Section 2.2 will explain the fundamentals of digital twins and their structure and data handling. Section 2.3 will introduce four types of digital twins and review their current applications in the context of CNC machining. Section 2.4 will summarise the challenges associated with the widespread implementation of digital twin technology for CNC machining. Section 5 will conclude on the findings of what has been researched for digital twins, and Section 6 will research what types of machining errors and the courses. Section 7 will summarise the knowledge gaps and what will be addressed in this thesis.

2.2 Fundamentals of Digital Twins

2.2.1 Digital Twin Structure

A digital twin requires three basic elements, i.e. a physical system, a virtual copy and their connections. A complete structure of a digital twin system is illustrated in Figure 2.1, which contains six parts [36]. They are:

- Live-state information, which is available in a timeframe that is as close to the underlying event as possible.
- Digital coupling, which is the coupling between the data source and the data processor.
- State which is the condition that the physical asset is in at any specific time.
- Physical asset/process, which is the physical entity that is being monitored by the digital twin.

- The virtual representation is a representation of the physical asset/process in virtual space through data or modelling.
- Functional Output – This is the information that is sent to the operator or back to the system that results in a change.

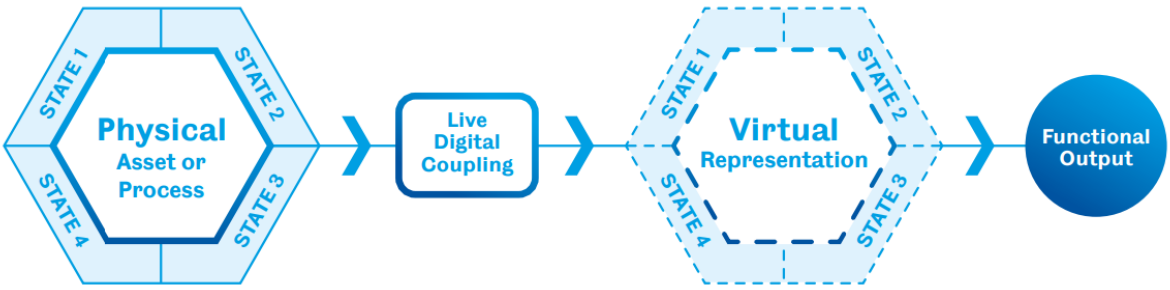


Figure 2.1 AMRC Digital Twin Structure [36]

2.3 Data handling in the digital twin

The digital twin can process data in three primary ways: data-driven, model-driven, or hybrid.

Data-driven models use input data from sensors or other sources to determine the correct response. They rely heavily on real-time data acquisition and analysis, making them highly effective when ample data is available and the physical system being modelled is relatively simple.

Model-driven models use a deep understanding of systems, often incorporating 3D, thermal, and kinematic models to determine the appropriate actions. These models are beneficial when the physical system is complex, and data availability is limited. They depend on theoretical and computational models to simulate the physical system's behaviour.

Hybrid models combine aspects of both data-driven and model-driven approaches to achieve a comprehensive understanding of the entire system. This integration allows for a more advanced digital twin but requires more computing power and greater system complexity. Hybrid models

leverage the strengths of both approaches to provide more accurate and reliable predictions and controls.

Choosing between a data-driven, model-driven, or hybrid approach depends on several factors, primarily the amount of available data and the complexity of the physical system that needs to be twinned [37]. For systems with abundant data and straightforward physical characteristics, a data-driven solution is typically optimal. Conversely, for highly complex systems with limited data, a model-driven solution is more appropriate [10] [38]. The choice is thus contingent upon the specific requirements and goals of the digital twin application [39]. Once the type and amount of data are known, it is crucial to develop a long-term data handling strategy. This strategy ensures the system's sustainability and scalability, preventing obsolescence and allowing room for future development and enhancement.

2.4 Digital twins for CNC machining

This section will further explain how digital twins used for CNC machining can be broken down into different types for easy categorisation, with examples of what they are being used for and the data structures used.

2.4.1 Types of digital twins for CNC machining

According to different functions, current digital twins for CNC machining can be classified into four categories, which are Supervisory, Interactive, Predictive and Composite Digital Twins[36]. Figure 2.2 illustrates these digital twins and how data is handled. The four types of digital twin are described as:

2.4.1.1 Supervisory digital twin

This type of digital twin displays information of a physical system or operation to an operator. This allows an operator oversight of the physical system, easing decision-making as they have access to up-to-date information of the whole system.

2.4.1.2 *Interactive digital twin*

This type of digital twin uses the data collected to apply feedback control of the physical assets from the virtual simulation for optimisation based on results from the data structure.

2.4.1.3 *Predictive digital twin*

This type of digital twin uses the data collected to predict the future physical state so corrective actions can be taken pre-emptively.

2.4.1.4 *Composite digital twin*

It is comprised of a network of digital twins that provide data to another digital twin, which is monitoring or controlling the network in some way. This is to allow a digital twin to be run at a low level, supervising a system, and one for fleet maintenance or monitoring.

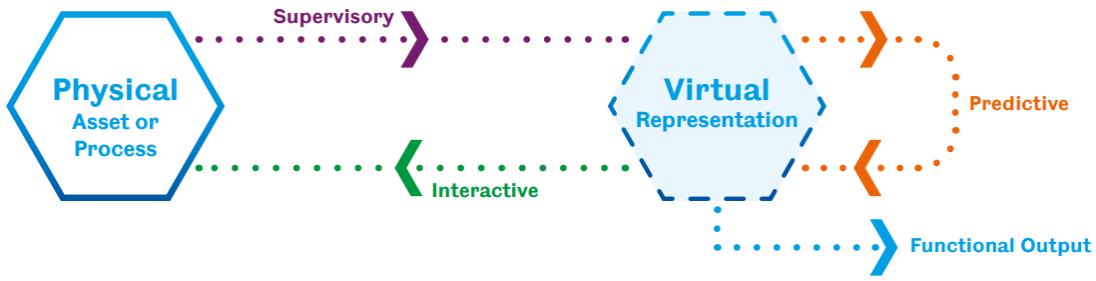


Figure 2.2 Four types of digital twins and data handling[36]

2.4.2 Supervisory digital twins for CNC machining

Examples of current research on supervisory digital twins illustrate their potential to enhance machine performance in various contexts. Below are some notable studies:

Botkina et al. used digital replicas of machine tools to provide details on wear, dimensions, and history through a modified ISO 13399 [40]. This ISO standard and machine status were updated using a tweeting machine and event-driven Line Information System Architecture (LISA). The approach enhanced data collection for operators, enabling better tracking of tool wear and machine performance, which could extend tool life by predicting issues. The researchers

encountered challenges in determining the exact tool dimensions, accuracy, and installing the tweeting machine for simulations. Tool data updates were facilitated by "ToolMaker" software developed by Sandvik Coromant and updated by KTH Royal Institute of Technology[15]. This method proposed new data collection techniques for production lines, potentially improving simulations of worn tools and identifying production rates and bottlenecks. However, the paper lacked test results, making the digital twin's effectiveness unclear, and noted that most CNC machines using PLC controllers could not install the tweeting device, complicating its application.

Xian developed a digital twin capable of simulating the machining process in real-time more accurately than existing methods [41]. The digital twin first read the STEP-NC file to understand the toolpath and then communicated with the CNC controller to synchronise tool locations. An optimised tri-dexel-based simulation simplified the mesh by reducing it into smaller sections for processing. This hybrid method combined data collected directly from G-code with CNC controller data to generate a new 3D model for simulation. Tested on 3-axis and 5-axis machines, the digital twin updated the virtual model every 50 milliseconds, faster than the control update, ensuring synchronisation. Future research would include adding physical status information for cutting forces, surface quality, and tool wear, along with integrating cloud-based monitoring.

These studies demonstrate how supervisory digital twins can enhance CNC machining performance while also highlighting the challenges of coupling physical and virtual components. The challenges faced by the paper reviewed are [40] faced difficulties in generating accurate machine tool 3D models, and [41] had to collect real-time tool position data and compare it to the toolpath. An additional challenge for supervisory digital twins is effective data visualisation, ensuring that information is presented clearly and is easily interpretable by machine operators. Poor visualisation can lead to misinterpretation of data and

result in incorrect operational decisions. Supervisory digital twins are sometimes confused with digital shadows; however, the key distinction lies in the direction of communication. Digital shadows only collect and visualise data, such as position or temperature, through one-way communication from the physical system to the digital representation. In contrast, supervisory digital twins enable two-way communication, allowing collected data to be analysed and used to inform decisions or send information back to the physical system or other connected systems. The CNC controller industry is addressing this challenge by developing high-precision controllers that can communicate with external devices and networks, thereby reducing data collection complexity. Advanced CNC controllers can perform error calculations and apply corrective actions like digital twin systems. However, these controllers are generally limited to the data obtained from their internal sensors and machine feedback systems. In contrast, a digital twin can integrate data from multiple external sources, such as wireless sensors, environmental monitoring devices, and simulation results, enabling a more comprehensive representation and analysis of the manufacturing process. In summary, supervisory digital twins hold significant potential for various machining processes. However, developing efficient methods to collect data quickly in a usable format is crucial for their success in CNC machining.

2.4.3 Interactive digital twin for CNC machining

Zhao developed a supervisory digital twin for a micro punching machine used in the production of LCD screen light guides [40]. The digital twin aimed to increase the accuracy of punched microdots' positions and formations by synchronising the virtual and physical spaces. This synchronisation was primarily managed by controlling and passing position data, supplemented with additional methods. The researchers faced the challenge of measuring punch hole quality and positioning, which they addressed by integrating a camera for image processing. Intermediate image checking proved more stable and reliable due to the sampling

rate. This capability to monitor punched hole quality during manufacturing enabled adjustments in punch power and hole positioning accuracy in real-time. The data-driven model monitored the X, Y positions and assessed punched hole quality, increasing punching speed from 20-40 dots/sec to 20-65 dots/sec and improving XYZ straightness from ±5 μm/200mm to ±2 μm/200mm.

Holland [42] researched and tested an interactive digital twin for controlling an electric cam's servo for motion control, aiming to enhance control and repeatability, making them more suitable for CNC machining. In this study, the interactive digital twin was utilised for trajectory planning, state monitoring, and precise control through multi-dimensional simulation software and virtual-real interaction. This required an in-depth understanding of the electric cam's electrical and mechanical properties, as well as the means to correct errors. The digital twin is operated by calculating the system's electrical and mechanical properties for trajectory planning based on non-uniform rational B-spline (NURBS) and using a 3D model for visual representation. When the cam moved, the actual result was compared to the predicted outcome, with any discrepancies tracked by an AI genetic algorithm proposed. The digital twin employed a hybrid data structure, integrating both data-driven and model-driven approaches for trajectory prediction, properties assessment, and movement representation. The effectiveness of this control method was tested by repeating an 8 mm motion every 60 ms for 200 cycles, demonstrating that the digital twin achieved repeatable, accurate control of the cam.

He [43] developed an interactive digital twin designed to regulate the output forces of a motor through a gearbox based on temperature changes. This system aimed to provide an automated corrective method for internal heat generation in control gearboxes. Achieving this required digitising the system to understand its thermal properties for Proportional-Integral-Derivative (PID) control. The digital twin is operated by first evaluating the system in real-time and then performing mechanical and thermal simulations using the collected data to identify optimal

operational parameters. These optimised operations were then sent to the controller, with the system continuously updating to maintain a constant state. This digital twin was primarily data-driven, calculating the mechanical and thermal properties of the system without relying on 3D modelling. The digital twin's structure can be seen in Figure 2-1. Its effectiveness was tested by encasing a gearbox in a heater to induce temperature changes. The results showed that the gearbox output stabilised after temperature changes, validating the digital twin's corrective capabilities.

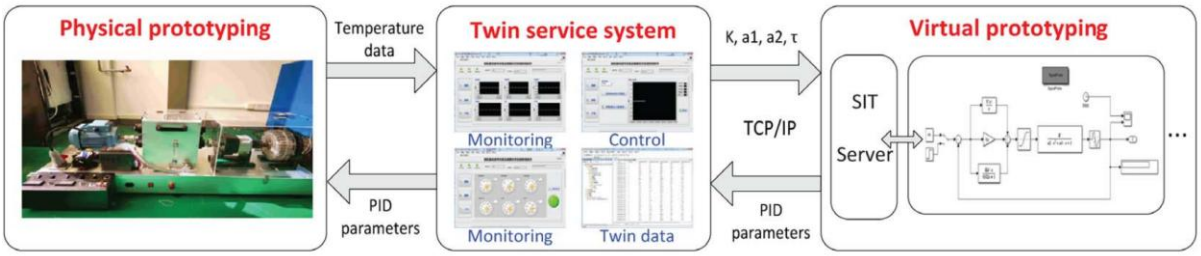


Figure 2-1 He's Tuning the Dynamic Digital Twin Structure [43]

Wang developed a digital twin to optimise clamping forces for thin-walled parts, addressing challenges of damage and low machining quality due to improper clamping [44]. Unlike traditional methods that rely on Finite Element Analysis (FEA) for simulations—which are computationally intensive and time-consuming—Wang's digital twin incorporates a neural network model capable of predicting appropriate clamping forces in real time. The digital twin framework integrates multiple components, including the physical clamping space, digital twin model, virtual clamping space, twin data, and interactions between physical and virtual spaces.

The neural network model was trained using data derived from FEA simulations. During operation, the model compares current conditions with historical data and the dimensions of the workpiece to optimise its behaviour, ensuring sufficient clamping force without causing damage. When tested, the digital twin achieved clamping force predictions within 3.41% of FEA results, demonstrating its effectiveness and accuracy.

The studies discussed in this section illustrate how interactive digital twins can be integrated into various systems to enhance performance. A consistent challenge in these implementations lies in identifying the critical data points required for the digital twin and validation of the trained model's accuracy for the application, preventing modelling errors. Collecting irrelevant or insufficient data significantly undermines the twins' ability to control the system effectively to predict accurate. For instance, [45] needed a method to measure punch hole accuracy. Holland [42] emphasises the importance of capturing the mechanical and electrical properties of cams and tracking their movements to adjust calculations accurately. Similarly, He [43] highlights the necessity of understanding the relationship between motor output and temperature, requiring precise detection of motor temperature and performance metrics to adjust output. Wang [44] underscores the need for accurate training data for AI models to determine optimal clamping forces without damaging workpieces.

Interactive digital twins function similarly to traditional feedback loops but differ in their complexity and the breadth of data points they can incorporate. As the volume and variety of collected data increase, the feedback loop's complexity can be enhanced, improving accuracy through machine learning and advanced data processing techniques.

2.4.4 Predictive digital twin for CNC machining

Examples of current research on predictive digital twins are given below to show how they can be used in different ways to improve CNC production capabilities.

Luo proposed a digital twin for predictive maintenance of a machine tool using multiple data structures to identify which would predict tool wear accurately [46]. This digital twin used multiple structures to identify the best approach, which was to collect data and process it to predict the Remaining Useful Life (RUL) of the machine tool with an AI algorithm. The data

points need to be identified, training models made for the AI of tool wear and identifying which approach is best.

The digital twin was validated by testing a cutting tool to failure with the wear measured intermetal. This data was used to train the digital twin for the model-driven prediction. The data-driven model used acceleration, cutting force and acoustic data to predict tool wear through the use of AI predictions. The hybrid-driven model uses a combination of data collected from both sources to determine the RUL of the tool. The results of the predictions are shown in Figure 2-2, where it is clear that the hybrid prediction matches the actual results best.

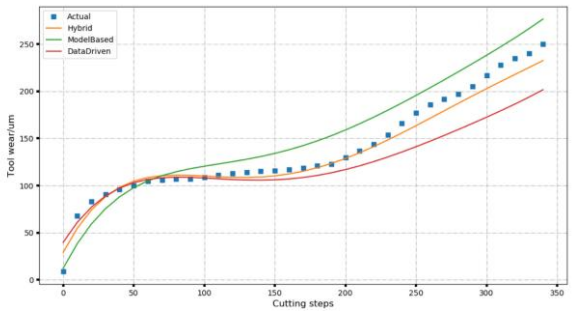


Figure 2-2 Comparison of Data, Model and Hybrid Driven Models [46]

The paper shows how different data models affect the accuracy of the predictions and the need for a tool for ware prediction digital twin, which could allow for greater accuracy in simulations of the machining process.

Hänel proposed a predictive digital twin for CNC manufacturing that would take historical data and UpToDate data from sensors to optimise the machining process, through simulations of processes and the use of a digital shadow to optimise the cutting process, machining quality and resource efficiency [47]. This requires all relevant data points to be identified, appropriate data structures for predictions and the connection between all of the systems. This was done by the creation of a digital shadow of the process, allowing for the collected data to be processed by the digital twin. The paper presents how it is believed the best way to collect, predict and display these results to the operator, each method uses a hybrid system that collects

the data to build the simulation that could be displayed to the user and tested virtually. These simulations were used to monitor, predict and correct toolpath inaccuracies, thermal behaviour of machine tool, material removal, cutting forces, recess stability, workpiece characteristics and surface quality. Figure 2-3 shows how toolpath inaccuracies are displayed to the user. It can be seen that the scope of this digital twin is very large, with well-researched ideas of the structure, calculations and required data. This system is very complex, and any small error between the physical and virtual systems could result in further errors in the predictions.

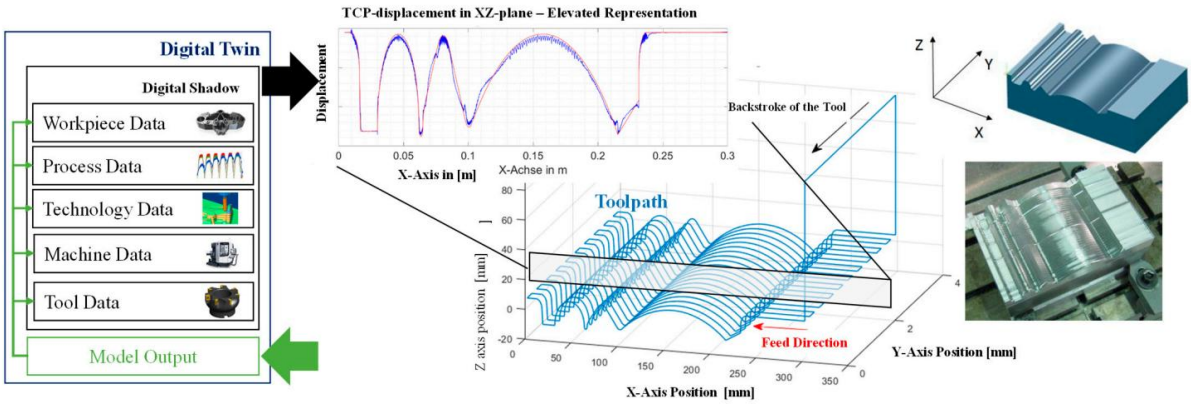


Figure 2-3 Updating Digital Twin[47]

Guo discussed how they developed a digital twin that could quickly predict collisions, stop the machine before a collision happens, display information about the CNC machine and be able to predict tool wear [48]. This was to prevent damage to the CNC machine and tool, as any collisions can be detected beforehand, and remote monitoring of CNC machines. Guo had to find a way to generate a virtual model of the CNC machine that could mimic the physical off and online, determine the most effective collision detection algorithm, have a system for remote monitoring of the machine and develop a fuzzy neural network (FNN) for tool wear prediction. This allows for collision detection to be applied to the machine while in operation in its current set-up without the use of CAM software for tool path simulation, allowing for collision prevention when the machine set up and the CAM set up are not calibrated the same. This system used a hybrid data structure with data-driven for collecting/sending, FNN tool wear

prediction from cutting forces and a model-driven approach for collision detection. This system was tested by turning parts of a marine diesel engine, where real-time data was displayed, visual reorientation, stop/start inputs and tool wear prediction that was accurate to 100% for a 25-micron error range and 95.2% for a 10-micron error range for a remote operator to use. The paper stated that more aspects of the tool wear prediction could be achieved by further development of this digital twin and that the algorithm could be improved further with more data sets.

Luo developed a digital twin that could self-monitor, maintain and predict [27]. This would allow for greater optimisation of machine usage, improved accuracy and greater reliability. To achieve this, the digital twin will need to have a method of data collection that can quickly be processed by the monitoring AI with historical data for predictive action to be taken. This process needs to be quick, meaning an efficient structure is needed as well. This digital twin only uses a data-driven model for predicting faults of the CNC machine. The effectiveness of this digital twin was tested on an X-axis where the ball screw nut position, speed, temperature and vibration were monitored, an FFN to predict failure using real-time data and historical data. The results of this production were found to provide failure prediction before it occurs. The paper stated that the digital twin can be expanded to the CNC system, and a method to apply a hybrid data structure for greater accuracy can be further developed.

This research gives examples of how a digital twin can be used to predict errors within a system and try to correct them. When developing a predictive digital twin, it is clear that the largest challenge is identifying what type of data structure to use, as it will determine how accurate the simulations are and how easily the models can be updated from the collected data while in operation. Luo[46] showed how the effectiveness of prediction by a digital twin depends on what type of data handling method was used, showing how critical it is to use the correct type. The digital twin was tested to predict the wear of a machine tool, and the case study showed

promising results. Hänel [47] proposed a general model applied to the CNC machines to monitor machine performance. However, this model would be very complex due to the number of features. Guo [48] provided a method for predicting collisions and optimising tool paths while monitoring tool were required an AI and simulations to work together. Luo [27] used AI and simulations to monitor, maintain and predict the wear of a machine axis. These examples show the complexity of applying a predictive digital twin to one part of a CNC system. More research is needed to develop a predictive digital twin that can handle the complexity needed to predict errors for all systems in a CNC machine.

2.4.5 Composite digital twins for CNC machining

The section shows how composite digital twins can be used to monitor workspaces to optimise production.

Zhang [49] developed a digital twin system for a manufacturing shop that would allow for monitoring of all the machines and parts while machining. This would allow for greater scheduling optimisation, precise control and a reliable forecast of production. The digital twin worked by having five lays the pass data between them. The five layers were a part layer which had models and parameters of all parts, a component layer which modelled part relationships and data models, a machine layer which provided models of full machines and energy/material usages, and the last layer was a shop-floor layer which monitors the data being passed to show relevant data and layout. To achieve this, all machines need to be digitised to allow for data collection, the connection between all machines and devices, as well as the network structure for communication. For such a complex system, a hybrid approach was used where data on tool/machine activities were sent for tracking and position, and a model-driven structure for visualisation of the shop floor. This was tested by using the digital twin to control a 3-SPS/S adjusting platform while monitoring kinematic and mechanical models. The research realised

that a better data fusion method between physical and virtual was needed, as currently, it was challenging due to set-up and data collection.

Gunes et al. [50,] et Juarez et al.[50, 51] created and tested digital twins on production lines for Renault (car gearbox) and MASA (aluminium airframe) with focuses on fleet monitoring, maintenance, and energy efficiency. This was done to test the system architecture that used a local monitoring system that collected raw data and processed it for selected Key Performance Indicators (KPI). These KPIs would then be passed up through the network for ease of monitoring and fleet comparison. For fleet maintenance, a score can be determined for each KPI, allowing for quick and easy comparison between different machines within the fleet and previous results of the machine to monitor drift. Using this system and combining it with virtual milling software, the entire process of planning, setup and machining was digitally twinned for both Renault and MASA.

The improvements found for MASA were a 10% reduction of machine tooling lead time, a 20% reduction in design and set-up of the new process, a 10% reduction in cycle time, a 5-10% reduction in the tooling cost and a reduction of machine downtime from failure[50].

Renault found that they had a 10% reduction in development time, 37% reduction of energy consumption for new tools, 11% reduction in the design and set up of new processes, 5-10% reduction in tooling costs and a 35% reduction in energy consumption of components [51].

These examples of composite digital twins show how the idea of networked machines and digital twins can improve the performance of CNC machines. The application of composite digital twins is focused more on the management of a workspace to optimise the flow of work and minimise losses. The biggest challenge for composite digital twins is the integration and digitalisation of all parts to allow for accurate enough monitoring.

Zhang [49] shows how a digital twin can be used to monitor a whole production workspace for optimised operation and identifying bottlenecking through the use of multiple layers. [50, 51] developed digital twins used to monitor and assist in the production planning of a machine bay to increase the production rate. As these types of digital twins are very complex and require many processes to be digitised, there are few papers on them.

The future of composite digital twins relies on their ability to be easily applied to different types of production lines, allowing manufacturers to increase their production rates. For this to happen, further research is needed for streamlined development and installation of digital twins into new areas.

2.4.6 Comparison of current digital twin research in CNC machining

Comparison of current digital twins in CNC machining is illustrated in Table 2-1 from the aspects of goals, challenges, data handling method, benefits and limitations.

Table 2-1 Comparison of current digital twins for CNC machining

Digital twin	**The goal of the digital twin**	**Challenges faced**	**Data Handling Method**	**Benefits and Limitations**
Supervisory	Monitor a system and provide the collected data to be processed by the operator or machine.	The coupling between the physical and the virtual system, as the accuracy and reliability of the data in real-time is critical [23, 24].	- Data-driven [37] - Hybrid driven [39] - Hybrid driven [40] - Hybrid driven [52]	Benefits: Allow a greater amount of data collection, which can lead to further optimisation. Limitations: How data can be collected from the existing system or by attaching new sensors, and how useful the data is due to reliability, accuracy and how fast the data can be collected for processing.
Interactive	Automate a feedback loop from the	The data points are being collected and	- Hybrid driven [41]	Benefits: Allow for automated feedback loops for corrective action based on historical or real-time data

	collected data of the physical system based on the digital twins' calculations.	used to identify critical calculations.	- Hybrid driven [42] - Data-driven [43] - Hybrid driven [44]	Limitations: The development of the feedback loop to process and decision-making of corrective action.
Predictive	Automatic control of a machining system through feed-forward/back control loops	Identifying relevant data points and processing the data in a correct method for accurate predictions	- Model-driven [46] - Hybrid driven [47] - Hybrid driven [48] - Data-driven [27] - Hybrid driven [53] - Data-driven [54]	Benefits: Feed-forward/back loops allow for corrective action before an error has occurred, allowing for much greater reliability and accuracy. Limitations: The complexity of the system, which often requires AI or ML to be developed and trained, prevents large-scale application due to computational load.
Composite Digital twin	Data collection and management of multiple systems for optimising total performance and bay maintenance	Integration and digitisation of all parts of a workspace to monitor	- Hybrid driven [49] - Hybrid driven [50] - Hybrid driven [51]	Benefits: With greater monitoring of the workshop floor based on KPI's, allowing for full production line optimisation and monitoring. Limitation: A very complex system to connect and identify relevant KPIs to be monitored leads to expensive development, set-up and testing.

It can be seen that the current category system allows a better definition of each digital twin and how they are used within the CNC machining system. The four categories presented are Supervisory, Interactive, Predictive and Composite Digital Twin. The examples given are

demonstrations of current research being conducted for manufacturing digital twins and how the category system is applied. From reviewing the examples, the application and use of different types of digital twins have been shown with supervisory being to monitor a system that can allow for greater control through optimisation of the data collected for greater simulation or corrective measures. Interactive digital twins can automate the feedback loop with the system to apply corrective measures to the system based on key data points. Predictive digital twins use feedforward and back loops to optimise the system by predicting possible errors based on previous data. Composite digital twins are used to monitor and optimise a production line through KPIs and simulations of production process challenges. The papers that have been reviewed here are just a small sample of the possible solutions for digital twinning, and more research is needed.

2.5 Summary of future research on digital twin for CNC machining

Parris stated that 90% of manufacturing companies were investing in predictive capabilities in the next 5 years, with 75% investing in machine learning and Process Digital Twins[55]. This shows a clear interest in the widespread application of digital twins for manufacturing. Future research focuses are suggested as follows:

- Definition of a digital twin for CNC machining
- Standardisation
- Flexible and reconfigurable digital twin
- Effective interaction between the digital twin and the machine operator
- Implementation of digital twins with an ultra-precision machining system for quality control.

The thesis will focus on the last research challenges to implement digital twins with an ultra-precision machining system for quality control. This literature review has examined the

application of digital twins in CNC manufacturing, highlighting that research focused specifically on ultra-precision manufacturing remains limited and requires further investigation. Ultra-precision machining presents several distinct challenges compared with conventional CNC machining, including heat generation associated with reduced cutting loads, significantly lower feed rates, tighter tolerance requirements, and the specialised material properties of workpieces. These factors necessitate different approaches and solutions, and further research is required to determine how effectively digital twin technologies can address the unique challenges of ultra-precision machining.

2.6 Review of machining errors

When measuring machine tool error, it can be categorised into several types, with the two primary types being quasi-static error and dynamic error. These errors are caused by different factors and therefore require different mitigation strategies. Quasi-static errors refer to errors that are present when the spindle or work surface is stationary. Dynamic errors, although not yet universally defined, generally refer to errors that occur due to the movement of the CNC machine or when the spindle is in operation [56-58]

The total machining error can be considered as the sum of quasi-static and dynamic errors. However, these are not the only forms of error that can occur during machining. As illustrated in Figure 2-4 by [59], for a five-axis CNC machine, multiple other error sources can also contribute to the overall accuracy loss; the same concept applies to machines with any number of axes.

The focus of this study is specifically on quasi-static and dynamic errors; therefore, the causes, effects, and mitigation strategies of other error types are not investigated at this stage.

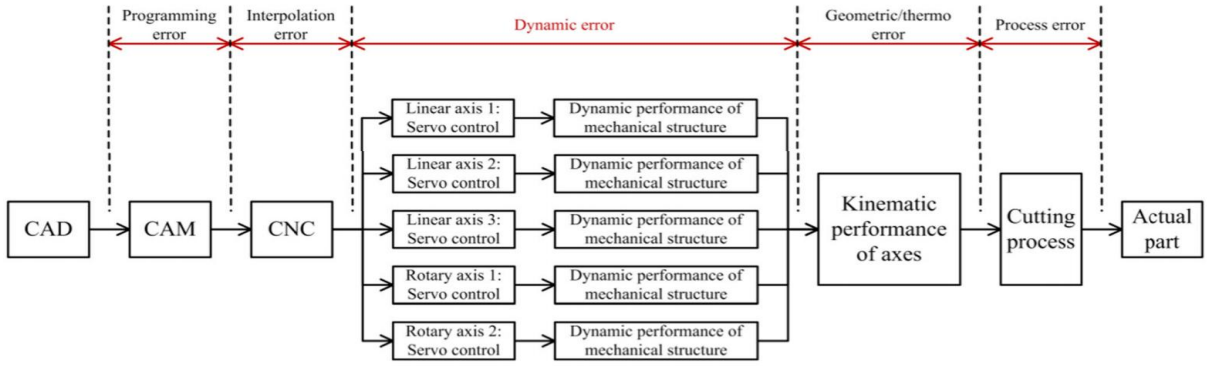


Figure 2-4 Types of Errors Within CNC Machines [59]

2.6.1 Quasi-Static Errors

As stated earlier, quasi-static errors are present when the spindle or workstation is not moving. This means that even at low machine speeds, these errors remain important to control and mitigate. Quasi-static errors can also vary over time, as the machine's geometric and kinematic characteristics change due to wear or damage [60]. This type of error can be further categorised into three main areas: geometric errors, static thermal errors, and gravity deformation errors [56, 61].

2.6.1.1 Geometric errors

Geometric errors, also referred to as kinematic errors, arise from imperfections in the geometry and dimensions of machine components [62]. These errors may originate from the design, manufacturing, assembly, or maintenance state of the machine. This aspect of quasi-static error focuses on the influence of the CNC machine's structural condition on the machining process. This relationship is illustrated in Figure 2-5 [63], where:

- shows what the ideal and desired movement would be,
- shows the errors that are present from manufactured parts, e.g., guides and ball screws,
- shows the errors that are present from the structure of the CNC machine from assembly and over time, e.g., wear of parts, gravity, stress release.
- shows the combined actual error from b) and c)

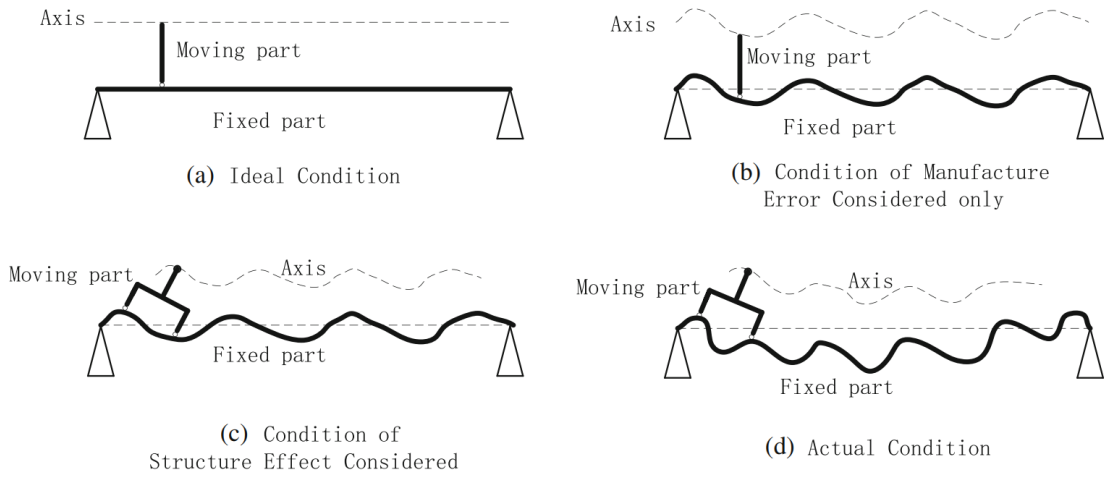


Figure 2-5 Geometric Error Illustration[63]

This actual error would be classified as the geometric error of the CNC machine and represents the deviation that must be measured, controlled, and mitigated to achieve accurate machining.

The most common solution involves incorporating encoders into the control axes of the CNC machine to monitor the degree of rotation. By accurately determining how much each axis has rotated, the CNC controller can calculate the corresponding movement of the tool head. Any discrepancies arising from incorrect axis rotation can then be detected and automatically corrected by the controller. However, this approach is limited by the accuracy and resolution of the encoder. To maintain precision, this approach is typically supplemented with regular maintenance to verify the straightness and squareness of the machine axes, ensuring that each axis moves precisely along its intended path, as well as using compensation tables. These tables enable software-based correction based on the known geometric errors of each axis. For example, squareness errors can be corrected by applying a proportional corrective movement between axes; for instance, a 1 m movement in the Y-axis may require a corresponding 0.01 mm adjustment in the X-axis. This relationship is determined either through machining tests

or by measurement using specialised instruments such as ball bars, laser interferometers, or precision artefacts.

This concept can be further extended to CNC volumetric compensation, where multiple geometric errors across the machine workspace are measured and corrected. For a three-axis machine, this typically involves compensating for 21 geometric errors, including six linear errors per axis and three squareness errors between the axes. As the number of machine axes increases, the number of geometric errors that must be compensated also increases, making the system more complex but also enabling higher levels of machining accuracy.

2.6.1.2 Thermal-mechanical errors

Thermal-mechanical error is related to the heating and uneven expansion of components within the CNC machine, which results in distortion of the machine structure[56], as illustrated in Figure 2-6. This thermal expansion occurs due to both internal and external heat sources[64]. When referring specifically to quasi-static thermal errors, it relates to external heat sources, as these are not generated by the CNC machine during operation[57]. External heat sources are associated with the environment in which the CNC machine operates and will vary between workshops. Common causes include ambient temperature, personnel or heat radiation, airflow, and solar intake. As environmental conditions fluctuate, thermal drift can occur during setup or operation, leading to unmeasurable errors that affect machining accuracy.

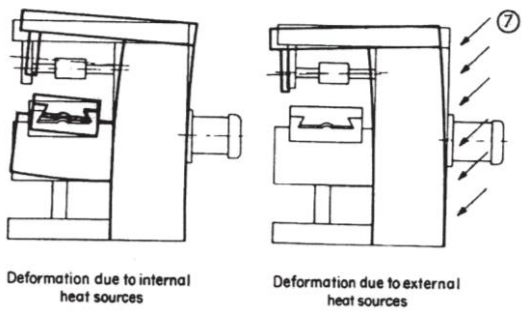


Figure 2-6 Effect of External Heat Source on Structure [65]

The quasi-static thermal error is compensated for through the creation of a data map that allows the prediction of errors in practical CNC machines. This method works by measuring critical points on the CNC machine and referencing them against the data map, which provides the necessary corrective offsets. The data map is typically created by the manufacturer using both simulation and experimental testing. Conducting a full thermal simulation of the CNC machine in real time would require excessive computing power and is therefore not feasible; consequently, this prediction-based compensation method is used instead[65].

2.6.1.3 Deformation of the workpiece due to gravity

This effect has a greater influence on the error of the tool tip than on the material itself, but it will still impact the overall accuracy of the finished part. The axis affected by this error typically exhibits a repeatable pattern, which can be approximated depending on the position and/or orientation of the machine [63, 66]. The extent of this effect will vary depending on the design, construction, and type of work being performed, meaning that testing is required to develop accurate compensation models.

This issue is commonly mitigated by employing appropriate security methods for both the tool and the workpiece. Such methods include the use of jigs, clamps, vices, step clamps, and toe clamps, which are among the most frequently used. When applied correctly, these methods help prevent workpiece deformation due to gravity, ensuring that it is securely held and properly supported during machining[67].

2.6.2 Dynamic Errors

The cause of the dynamic error is directly related to the movement of the CNC machine. As the feed rate increases, the dynamic forces acting on the machine also rise, leading to greater potential for error that must be prevented, mitigated, or corrected. These dynamic effects

typically manifest in the form of vibration, thermal expansion, jerk, and tool path trajectory deviations[56, 61, 68, 69].

In CNC machining, dynamic error is the deviation between the tool and the reference (setpoint) displacements. From a state-of-the-art review of the dynamic error of CNC machine tools conducted by [56] It has been found that the dynamic error can be split into two different categories, which are the dynamic error inside the servo loop and the dynamic error outside the servo loop.

2.6.2.1 dynamic error inside the servo loop

This phenomenon is commonly referred to as tracking error, which occurs when the system cannot maintain an accurate measurement of the tool or axis position relative to the intended position. In a fully closed-loop servo feed system, the tracking error primarily arises from the linear scale, whereas in a semi-closed-loop system, it originates from the motor encoder. These differences are illustrated in Figure 2-7. Examples of such errors include insufficient resolution of linear or rotary encoders, which introduce play in the system and result in discrepancies between the actual and intended positions.

2.6.2.2 dynamic error outside the servo loop

This error represents the difference between the actual end effector or tool tip position and the position measured by the encoders. This component of dynamic error increases with higher feed rates and machining speeds due to the greater forces and accelerations involved. For example, doubling the velocity requires a fourfold increase in acceleration. The resulting inertial forces during rapid machining cause elastic deformation of the end effector or tool tip, which cannot be measured by the encoders since it occurs outside the control loop; this phenomenon is commonly referred to as jerk [70]. Jerk can also induce vibrations within the

CNC machine structure, further increasing the displacement between the actual and sensed positions, as these effects are typically not captured by the control system.

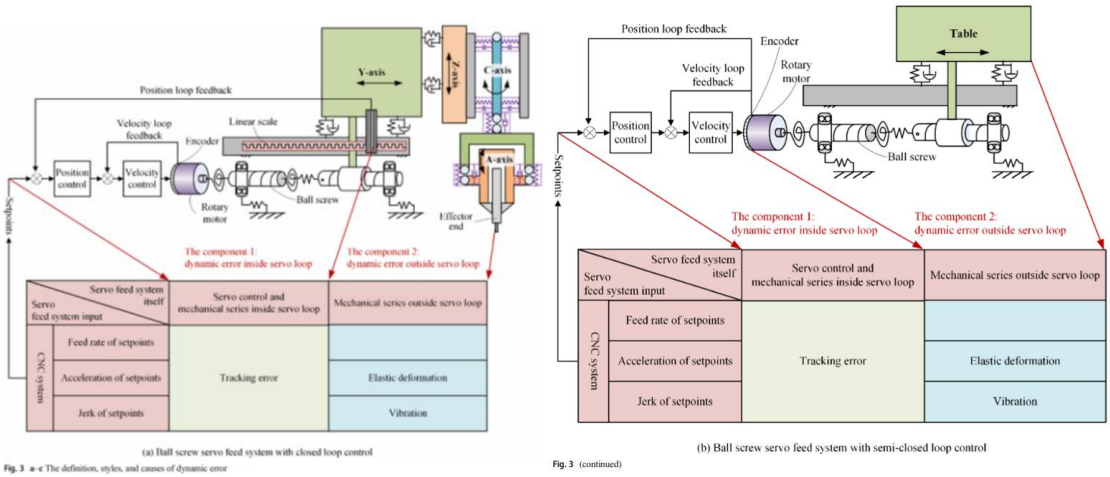


Figure 2-7 Dynamic Errors Inside and Outside of the Servo Loop[56] (Left is a closed loop, right is a semi-closed loop)

2.7 Ultra-precision machining

Ultra-precision machining generally refers to manufacturing processes that require dimensional tolerances and surface finishes at the nanometre scale. This level of accuracy enables significant advancements in production methods, such as achieving smooth surfaces in hard and soft contact moulds without the need for subsequent polishing [71], or fabricating microstructures on materials to impart new or enhanced properties, including super hydrophobicity, self-cleaning, drag reduction, improved lubrication, wear resistance, and energy conversion [72].

There are four main types of ultra-precision machines: turning, milling, scratching, grinding, and polishing. For this thesis, the focus will be on ultra-precision milling, with particular emphasis on ball-end milling or ultra-precision high-speed milling, as this is the type of machine employed in the experimental work.

The challenges of ultra-precision machining are fundamentally the same as those of conventional machining, with errors originating from similar sources; however, the measured

data are of a much smaller scale. These smaller error signals require more complex and sensitive measurement devices, which are inherently more susceptible to sudden changes in position, impacts, and environmental disturbances. Due to the difficulty of measuring such small signals reliably and to prevent tool damage, slow feed rates are typically employed in ultra-precision machining processes.

This results in longer production times, with up to 20% of the total production time being spent on machining complex parts alone [18], leading to high production costs [73]. Production time is further increased by lengthy setup operations; in traditional machining, machine setup can account for 40–50% of the total production time [74].

The use of digital twins has the potential to reduce production time and improve machining accuracy. It has been estimated that up to 40% of the production time involved in mould manufacturing could be reduced through the application of intelligent manufacturing systems [74]. To address these challenges, digital twins are expected to play a critical role in the future of ultra-precision machining processes [73].

Several areas have been investigated where digital twins are applied to smart control for ultra-precision machining. Xu presents a study in which a digital twin is used to track errors between commanded and actual positions using deep learning to improve machining precision [75]. This approach employs a G-code interpreter combined with a deep learning model based on TCN-BiLSTM (Temporal Convolutional Network and Bidirectional LSTM) to predict X- and Y-axis tracking errors in real time. The reported results show a tracking error reduction of 45–75% for the X-axis and 40–70% for the Y-axis.

presents a digital twin that autonomously controls a CNC mill for the manufacture of thin-walled aerospace components. The digital twin integrates real-time machine data with simulation modelling to monitor and control wall deformation during machining [76]. This

approach was shown to minimise machining deformation with minimal lag compared to conventional methods.

Similarly, presents a digital twin that provides adaptive control based on cutting forces during milling and addresses limitations in traditional methods when cutting conditions change significantly [77]. The digital twin predicts cutting forces using real-time data derived from feed-drive motor currents and adapts the feed rate to maintain cutting forces near a desired threshold. This approach was shown to improve machining performance and surface quality.

Overall, digital twins have demonstrated the ability to reduce errors during real-time machining through automated corrective actions. With accurate and reliable error correction, machining feed rates can be increased without compromising part quality, thereby reducing production time and manufacturing cost. Production efficiency can be further improved through automated setup processes, as demonstrated in [74], where the real machine setup time was reduced by 40%.

This approach can be extended to ultra-precision machining through the use of robotic arms for automated workpiece handling. A robot equipped with its own corrective-action digital twin could perform loading and unloading operations with high accuracy, reducing setup time and increasing production rates. This concept was investigated in [78], where a roughly machined part was positioned on an ultra-precision machine for final finishing passes. The loading process was performed by an industrial robot, with placement errors corrected using numerical control. Compared to machining the entire part on the ultra-precision machine, this approach reduced machining time by 76%, from 40 hours to 9.5 hours.

Such automated workpiece handling systems also enable the possibility of lights-out manufacturing for ultra-precision components, as the entire process from loading and machining to unloading can be fully automated.

2.8 Summary

This chapter reviews the state of the art in digital twin technology for CNC machining, machining errors, and the specific challenges of ultra-precision manufacturing. Digital twins are presented as a key enabler of smart manufacturing, providing a live virtual representation of physical systems that supports monitoring, simulation, prediction, and control. The fundamental structure of a digital twin is defined through the integration of physical assets, virtual models, real-time data coupling, and functional outputs.

Digital twins are categorised according to their function into four types: supervisory, interactive, predictive, and composite digital twins. Supervisory digital twins focus on monitoring and visualisation, interactive digital twins enable real-time feedback control, predictive digital twins anticipate future system states for feedforward correction, and composite digital twins integrate multiple digital twins to optimise entire production systems. Examples from recent research demonstrate the benefits of each type while highlighting common challenges, particularly data acquisition, data handling, system coupling, and computational complexity.

The chapter also reviews machining errors, classifying them into quasi-static and dynamic errors. Quasi-static errors include geometric errors, thermal-mechanical errors, and gravity-induced deformation, which are typically compensated through encoders, data maps, and fixturing. Dynamic errors arise from machine motion and are further divided into errors inside the servo loop (tracking errors) and errors outside the servo loop, such as elastic deformation and jerk, which cannot be measured by standard encoders and become more significant at higher feed rates.

Finally, the chapter introduces ultra-precision machining, which requires nanometre-scale accuracy and presents additional challenges due to sensitivity to dynamic errors, thermal

effects, and workpiece handling. While digital twins have been increasingly applied to CNC machining and precision control, gaps remain in their application to ultra-precision manufacturing, particularly in real-time dynamic error compensation and automated workpiece handling for production rate reduction. These identified gaps provide the motivation and foundation for the research presented in this thesis.

The knowledge gaps in the study of the digital twin-driven ultra-precision machining system are identified as follows:

- Lack of an effective in-line dynamic error measurement approach for an ultra-precision machining system
- Lack of a real-time digital twin model of dynamic errors for an ultra-precision machining system.
- Lack of a digital twin-driven dynamic error compensation approach for an ultra-precision machining system
- Lack of a digital twin framework capable of seamless integration of multiple digital twins for automated processing.

The thesis will develop a digital twin-driven ultra-precision machining system to bridge the above knowledge gaps.

3 Chapter 3 – Research Methodology

3.1 Introduction

This chapter explains the methodology used to develop a digital twin–driven ultra-precision machining system for manufacturing. The proposed system consists of a higher-level digital twin that coordinates two subsystem digital twins: a COBOT and a hybrid mill. Together, these enable an automated manufacturing system capable of producing ultra-precision components more efficiently. By automating loading and unloading operations and synchronising real-time corrective control, the system aims to reduce overall production time while maintaining high tolerance levels.

The COBOT digital twin connects to and controls the robot by using measured dynamic error data to modify the programmed tool path and apply corrective actions in real time. For this research, real-time refers to the application of corrective action when predicting system error. The operational speed of the system must be as fast as possible to ensure that the predicted corrective action remains relevant. Consequently, the system must update its corrections within an appropriate time scale. As this represents a novel method of error measurement and correction, real-time performance in this instance is defined as multiple updates per second, with a target minimum update rate of 3 Hz (equivalent to an update delay of approximately 300 ms). Implementing such corrections during operation is challenging, as the motion of each joint must be recalculated dynamically. As a result, corrective actions are typically applied offline; this research seeks to demonstrate an effective real-time solution.

The CNC digital twin applies real-time corrective actions to the hybrid mill based on measured dynamic errors. This requires an approach that does not modify the programmed G-code, as such changes can interfere with the CNC controller's internal trajectory calculations.

Conventionally, corrections are applied offline or by reducing machining speeds to minimise dynamic errors. The methodology presented in this chapter aims to overcome these limitations.

3.2 Digital twin-driven ultra-precision machining system

3.2.1 Ultra-precision digital twin composite structure

As stated, this digital twin will be composed of three components: one digital twin for each machine and a higher-level digital twin responsible for overall control, data collection, and communication between all devices. The structure and responsibilities of the proposed composite digital twin are illustrated in Figure 3-1.

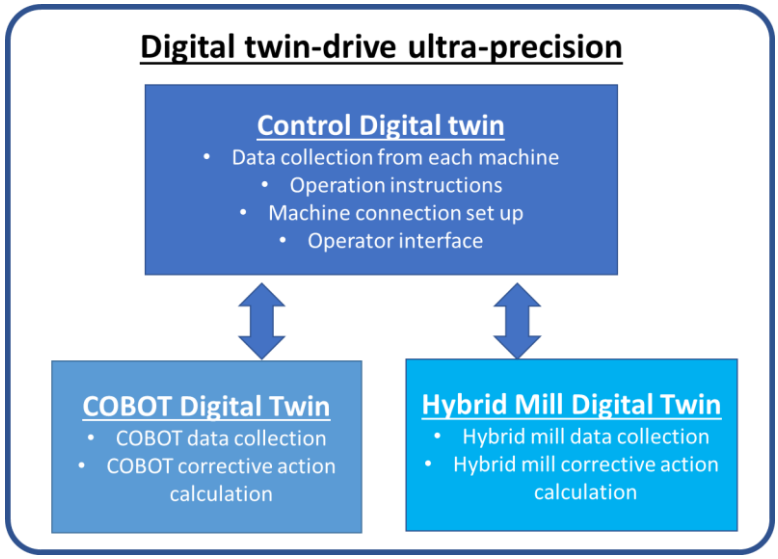


Figure 3-1 Structure of digital twin

As shown in Figure 3-1, the control digital twin is responsible for the following functions:

- Collecting data from both machine-level digital twins and structuring and storing it in a usable format for later analysis and reuse.
- Managing operational scheduling by determining which digital twin should be active at a given time to reduce computational load.
- Coordinating the connection and interaction between the COBOT and the hybrid mill to minimise downtime.

- Providing process status information to the operator.

Overall, the control digital twin is responsible for the coordinated and reliable operation of the two machine-level digital twins, enabling them to work together in an automated manner to reduce the total production time of the complete system.

3.2.2 Predictive digital twin-drive dynamic error compensation approach

The proposed approach applies corrective action by extending the feedback control loop with an additional feedback loop from accelerometers in the form of a dynamic error correction term. This correction is derived from acceleration data measured near the tool centre point, enabling errors outside the conventional control loop to be captured and compensated. The resulting digital twin–driven dynamic error control loop is illustrated in Figure 3-2.

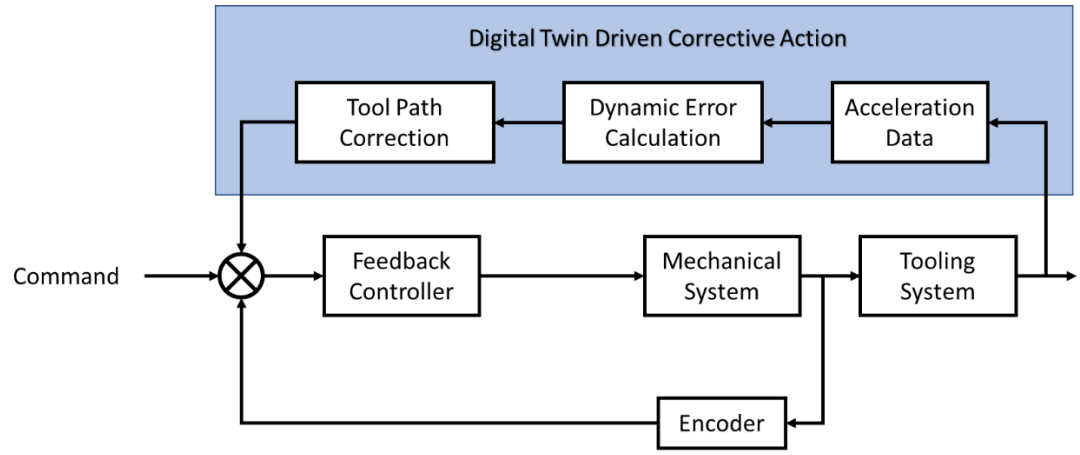


Figure 3-2 Extended digital twin–driven dynamic error control loop

This thesis will develop a digital twin system for ultra-precision machining. The system includes one main digital twin that coordinates two smaller digital twins: one for a COBOT and one for a hybrid mill. The goal is to automate the process, reduce production time, and maintain machining tolerances.

Both machines use accelerometers to measure dynamic errors that cannot be detected by standard encoder control loops. All sensor and machine data are sent to a central PC, where the

digital twin processes the data and calculates real-time corrective actions. The system also uses stored machine information, such as G-code and machine settings, to support the corrections.

Dynamic error correction is achieved by improving the normal feedback control loop using accelerometer data. Two methods are used: calculating displacement through double integration of the acceleration signal and using machine learning to predict dynamic errors directly for compensation.

The use of double integration to obtain displacement from acceleration presents several challenges, as accelerometers detect a noticeable amount of noise. This noise is further amplified through double integration, requiring a filtering process to remove it in order to achieve accurate displacement calculations. The use of accelerometers also presents challenges regarding the measurable frequency range of the sensor, meaning that correct sensor selection is critical for this application.

Although the correction methods differ between the COBOT and the hybrid mill due to differences in control systems, the overall approach is designed to be adaptable to other machines.

The resonant frequency of the COBOT and the hybrid CNC machine is 10 Hz on average within the work envelope and 194 Hz, respectively, based on a tap test carried out by other colleagues in your team.

3.2.3 Dynamic error measurement sensor selection

The use of an accelerometer has been considered over other measurement methods as there are positive and negative reasons for its selection; these reasons and other options considered are presented below.

3.2.3.1 Accelerometers sensor advantages and disadvantages

Advantages

- Small, lightweight sensors that can be easily attached to machine structures.
- Do not require line-of-sight operation with other devices.
- Relatively insensitive to many external environmental changes.
- Cost-effective compared with many high-precision metrology systems.
- Available in a wide range of measurement ranges and sensitivities.

Disadvantages

- Sensitive to external vibrations that are not related to the measured motion.
- Signal filtering is required to reduce noise and prevent drift over time.
- Displacement measurement requires additional calculations (e.g., integration), which may introduce further errors.

3.2.3.2 Laser interferometer sensor advantages and disadvantages

Advantages

- Capable of sub-micrometre tolerance distance measurements.
- Environmental influences can be compensated for through calibration sensors.
- Able to measure displacement errors directly over large distances.
- Can measure multiple types of machine tool errors and is widely regarded as an industry-standard calibration method.

Disadvantages

- The cost of the sensor system and associated equipment is very high.
- Requires a clear line of sight between the laser source and the reflector.

- Requires a relatively long start-up and setup process and is sensitive to laser misalignment.
- Typically, not suitable for continuous in-process monitoring of machine tools.
- Reflective optics required for operation can be bulky and may restrict machine operation.
- Performance can be affected by airborne contaminants such as dust, oil, and coolant mist.

3.2.3.3 Capacitive sensors: advantages and disadvantages

Advantages

- Capable of sub-micrometre measurement accuracy with high precision.
- High sampling rates allow rapid response to displacement changes.
- Do not require line-of-sight operation.
- Sensor probes are small and lightweight, enabling integration in compact setups.

Disadvantages

- Very limited measurement range when operating at high precision (typically <5 mm).
- Sensitive to environmental contamination such as dust, oil, and coolant.
- Requires precise setup and alignment to maintain the correct sensing gap.

3.2.3.4 Digital twin-driven selected sensor

From the three sensors considered for measuring the dynamic error of both machines to develop a data-driven digital twin, the accelerometer was selected. This approach has not been widely studied for this application and therefore has the potential to provide a novel method for measuring dynamic machine error in manufacturing systems.

The accelerometer also offers a more universal solution, as it requires only a small mounting area on the machine and a cable connection to the data logger. Due to its small size and low mass, it is unlikely to influence machine operation, unlike systems such as laser interferometers, while maintaining similar integration advantages to capacitive sensors.

In addition, accelerometers can operate at any orientation, with alignment corrections applied through software. They also do not require additional reference components for measurement, unlike capacitive sensors, which must maintain a fixed sensing distance from the measured surface; deviations from this distance may introduce additional measurement errors.

The selection of accelerometers for this task, therefore, provides a potentially novel and cost-effective solution that could be applied universally across different machines, enabling further research into dynamic error measurement and compensation.

3.2.4 Concept of digital twin-drive ultra-precision machining system

With the outline of the digital twin–driven ultra-precision manufacturing system established, further development is required to define the key functional areas. A conceptual connection diagram has been created, as shown in Figure 3-3, which illustrates the necessary components of the overall digital twin and their interconnections.

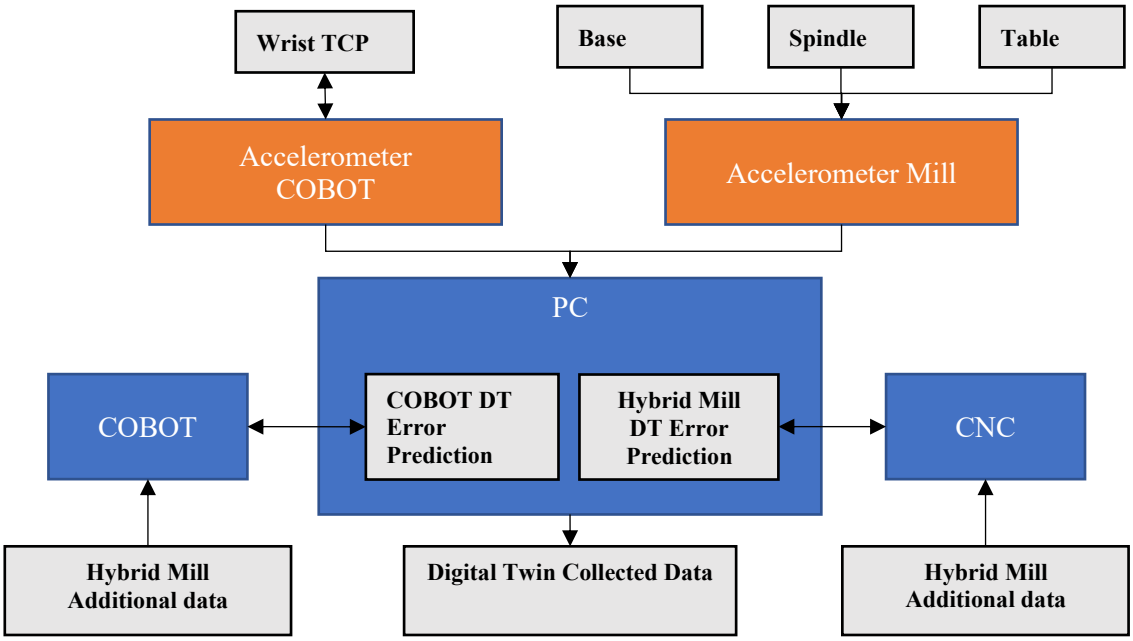


Figure 3-3 Conceptual digital twin connection and components

Both machines are equipped with dedicated accelerometers. For the COBOT, an accelerometer is mounted at the wrist near the tool centre point (TCP), while for the hybrid mill, accelerometers are installed on the table, spindle, and base. The base-mounted accelerometers are used to measure external vibrations affecting both machines, allowing these influences to be identified and removed from the measured signals. The data to be collected from each machine's digital twin will be:

COBOT

- Acceleration Wrist (XYZ)
- Acceleration Base (XYZ)
- COBOT TCP reference to base (XYZ)

Hybrid Mill

- Acceleration Table (XYZ)
- Acceleration Spindle (XYZ)
- Acceleration Frame (XYZ)

- Position Feedback (XYZ)
- Position Command (XYZ)
- Position Error (XYZ)
- Axis Average Current Draw (XYZ)
- Axis Average Velocity (XYZ)
- Feed rate
- Manual Feed Rate Override
- Spindel Speed

All sensor data and machine-accessible data are collected and processed on a central PC, where the digital twin performs data processing and determines the required corrective actions. These corrections are then transmitted back to the machines. In addition to real-time data, static machine information is stored within the digital twin for reference, including workpiece material properties, G-code, and machine acceleration characteristics.

3.3 Summary

This chapter presents the research methodology for a digital twin–driven corrective action framework for ultra-precision manufacturing. It outlines the proposed system architecture, which comprises individual digital twins for the COBOT and the hybrid mill, each tailored to the specific performance characteristics and corrective action requirements of the respective machine, alongside a third control digital twin responsible for data collection, coordination, and system operation.

Corrective action is based on errors measured outside the conventional control loop using accelerometers attached to both the COBOT and the hybrid mill. The measured acceleration data are used to calculate dynamic errors, which are then compensated for in real time. Two

methods are employed to determine dynamic error: double integration of the acceleration signals to estimate displacement, and machine learning models to predict the error directly.

This approach extends the conventional feedback control loop by introducing an additional loop that incorporates external measurements, enabling the correction of errors that cannot be detected by standard encoder-based control systems. Overall system operation is managed by the control digital twin, which collects and stores data from each machine-level digital twin, determines when each should be active to manage computational load, and provides process status information to the operator.

This chapter has outlined the operational framework of the proposed digital twin, including its system architecture, the data to be collected, and the selection of the most suitable sensor for measuring errors occurring outside the machine control loop. Establishing these elements provides the foundation for the subsequent development of a working data-driven digital twin.

Using the knowledge gained from the literature review presented in Chapter 2, the development of two predictive digital twins, one for each machine controlled by a higher-level digital twin, is proposed as an effective approach. This structure enables coordinated control of both machines while also providing a clear interface for human interaction. In addition, it supports the implementation of corrective actions that function as an external feedback mechanism to the existing control system, enabling a potentially universal approach to dynamic error compensation.

This framework addresses the second research question concerning the most effective method for measuring dynamic errors outside the machine control loop, allowing the proposed approach to be experimentally evaluated. The overall effectiveness of the system must therefore be assessed through testing on each machine using its respective digital twin to determine its performance in practical applications.

4 Measurement Approach of Dynamic Errors of COBOT

4.1 Introduction

Chapter 4 will introduce a measurement approach which uses accelerometers to measure dynamic error outside the servo loop through double integration. The approach will be validated using a ball-bar test by comparing the calculated average radial error against the ball-bar results, which are treated as the ground truth.

In this measurement approach, dynamic error outside the control loop is obtained using accelerometers through measurement of displacement at the TCP in X, Y and Z three dimensions, thereby determining dynamic error in the XYZ axes. Displacement is obtained by double integrating the acquired acceleration signals with respect to time. The first integration is shown in equation 4-1 to get velocity, and the double integration shown in equation 4-2 is to obtain displacement.

$$v(t) = \int a(t)dt + C_1 \tag{4-1}$$

$$x(t) = \int v(t)dt = \int(\int a(t)dt + C_1)dt + C_2 \tag{4-2}$$

Using this method, the dynamic error of the machine can be measured at the location of the accelerometer. This approach will be applied to a COBOT and a CNC mill to demonstrate its universal applicability.

4.2 Experimental setup for measurement of COBOT dynamic errors

In this study, accelerometers, which are traditionally employed for vibration and chatter analysis [79-81], tool and machine vibration monitoring[82, 83], cutting force measurement[84, 85], and fault detection[86], are used to measure the displacement at TCP through double integration of acceleration data.

The setup of dynamic error measurement on a COBOT (UR10e) is shown in Figure 4-1 Two triaxial accelerometers (PCB model 356B18), with a bandwidth of 0.5 to 3000 Hz (±5%), are used in this study. One accelerometer is mounted at the wrist of the COBOT, while the other one is positioned at the base of the COBOT on the frame. The wrist-mounted accelerometer is secured using a steel frame that is attached to the wrist using two bolts. The accelerometers are aligned with the COBOT axes at the wrist using two bolts positioned off-centre from the centre of the accelerometers. This arrangement allows the axis alignment to be calibrated and secured from behind. The base accelerometer is secured and aligned with the base using the groove of the extruded aluminium frame and is bolted in place. Guide walls are incorporated into the printed mount to prevent rotational error between the accelerometer and the mount. The mounts for the accelerometer attached to the COBOT frame are 3D printed from Polylactic Acid (PLA). The orientation of the two accelerometers is kept the same, allowing for easier combination of the data. By subtracting the readings from the frame accelerometer from the wrist, the influence of background noise can be eliminated. The accelerometer data is collected through a data logger (National Instruments cDAQ-9174) and transmitted to a data acquisition card (NI-9234).

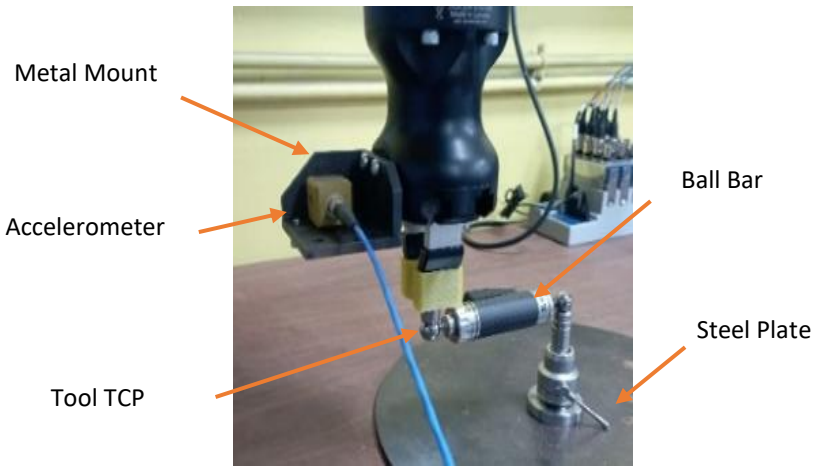


Figure 4-1 COBOT end effector set up

The use of the ball bar will be used to measure the radial distance from the centre point and will provide data to allow for the calibration and validation of the measurements of the COBOT to measure the effectiveness of the digital twin.

4.3 Measurement implementation

4.3.1 Data communication and operation procedure

This method utilises Cartesian positioning data since COBOT encoders provide joint angles. The use of Cartesian coordinates ensures the system's universality, as it remains independent of the machine's configuration. The only requirement is that the accelerometer is positioned on the same section as the TCP to minimise errors.

MATLAB was used for data handling and communication, while RoboDK software was employed for COBOT control. The system connection and data flow are illustrated in Figure 4-2.

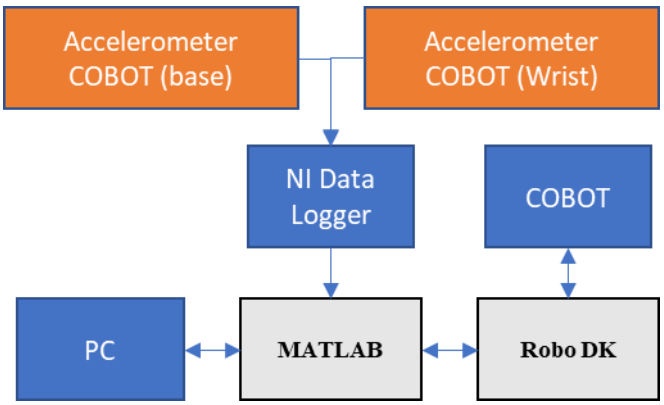


Figure 4-2 Data handling and communication

When the system is in operation, it loads the required toolpath programs and repeats the following process until program completion:

- Real-time positioning data is collected, along with acceleration values from both the wrist TCP and the base, with appropriate offsets applied to each.

- The wrist and base accelerometer data are filtered using Fast Fourier Transform (FFT) to remove noise.
- Vibrations measured by the base accelerometer are subtracted from those measured at the wrist to eliminate background noise.
- Acceleration values at the Tool TCP are converted to mm/s^2.
- Double integration is performed using cumulative trapezoidal numerical integration to convert acceleration data into displacement along the X, Y, and Z axes.
- Homogeneous transformation is applied to convert coordinates from the accelerometer TCP to the wrist TCP, and subsequently from the wrist TCP to the Tool TCP.
- Both the COBOT positioning data and digital twin data are stored for further analysis.

During program execution, real-time data and a corresponding visual representation are displayed to the user. Figure 4-3 illustrates the overall operation of the tracking system.

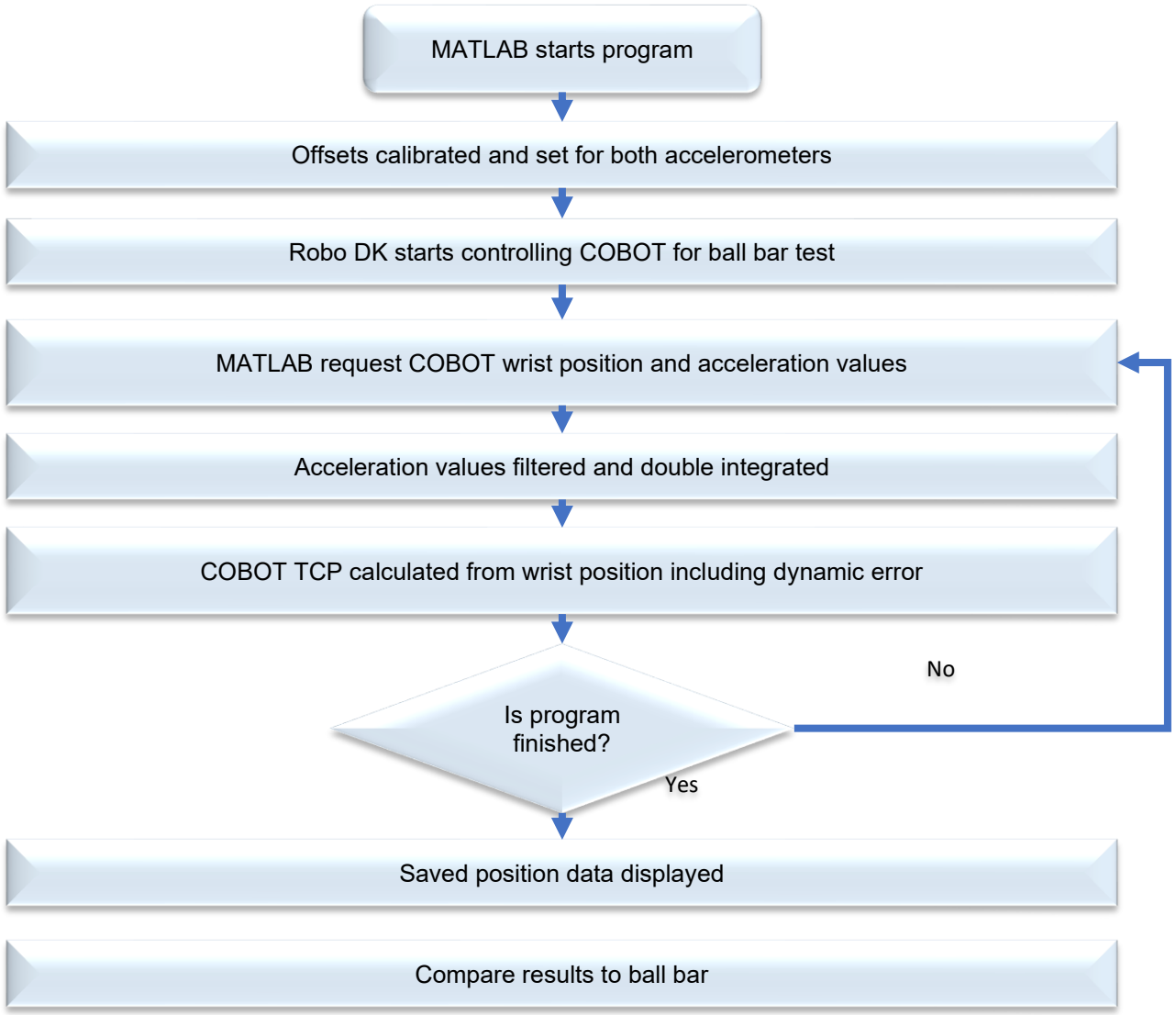


Figure 4-3 Operation of the tracking system

4.3.2 Data processing

As previously mentioned, displacement is determined by the double integration of acceleration over time. This calculation will be performed using MATLAB, where acceleration data will be collected, filtered, and double-integrated to obtain positional information. The methodology involves tracking the position of the accelerometer through precise displacement calculations. The process of this calculation is shown in Figure 4-4.

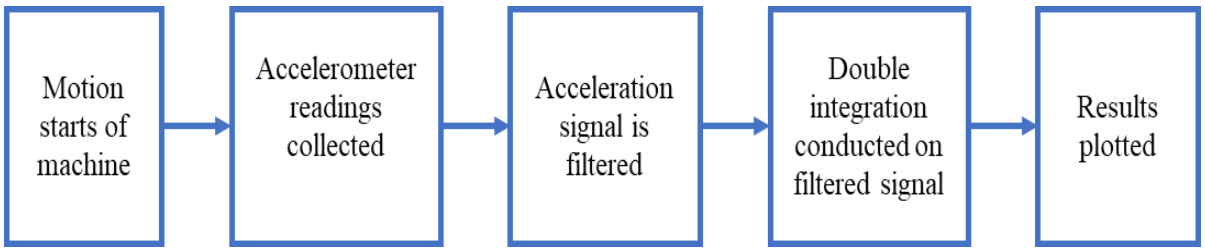


Figure 4-4 Methodology of displacement calculation

Fast Fourier Transformation was used as the filtering method, as it allows for real-time signal processing[87]. The process of FFT works by:

- The signal is represented as a series of discrete time-domain samples.
- The FFT divides the signal into smaller segments by exploiting the symmetry and periodicity properties of the Fourier Transform.
- These smaller segments are repeatedly processed and combined to generate the frequency components of the signal.
- The result is a set of complex numbers representing the amplitudes and phases of the signal's frequency components.

This process enables the frequency components of the signal to be visualised, allowing a cut-off value to be applied to remove components that occur infrequently. The cut-off value is a threshold that requires frequencies to appear in the signal more frequently than the defined limit. If a specific frequency does not appear often enough, it is removed from the signal. Unlike high-pass, low-pass, or band-pass filters, this filtering method does not impose a fixed frequency boundary that frequencies must be above or below. Instead, it allows frequencies that consistently occur within the signal to remain while removing those that appear only sporadically. As random noise typically does not occur consistently, this approach effectively eliminates it from the signal.

The appropriate selection of the cut-off value is critical, as it determines the boundary between useful data and noise. If the cut-off value is set too high, relevant information may be lost; conversely, if it is set too low, noise will remain in the filtered data. Figure 4-5 below shows this effect by showing the same signal with different cut-off values applied.

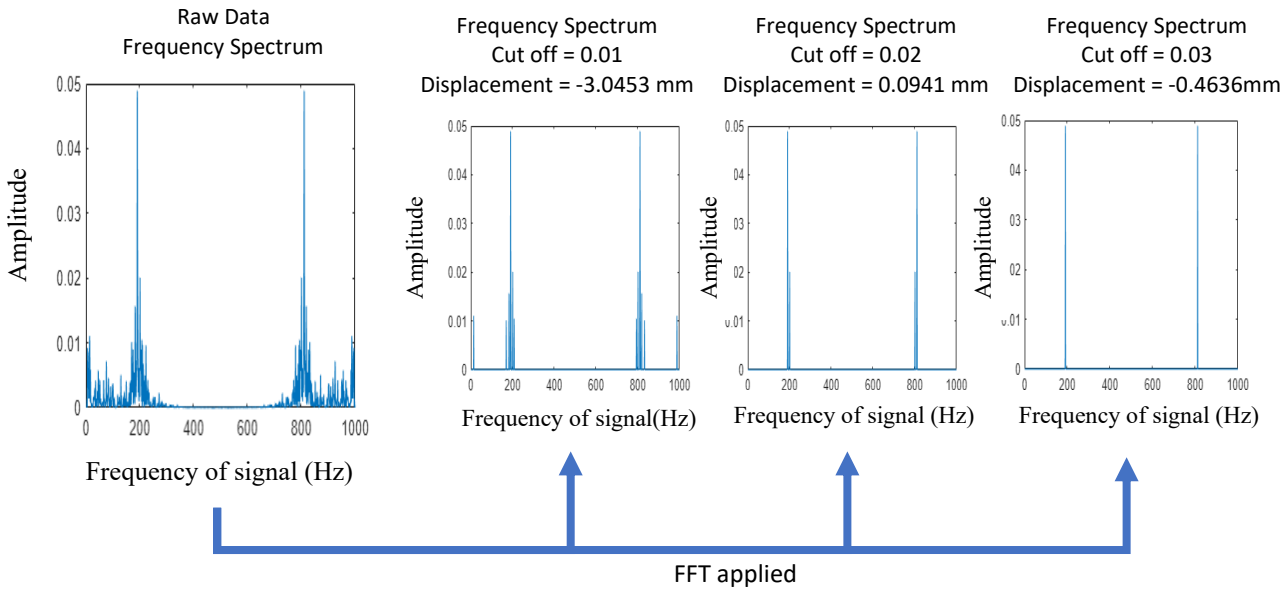


Figure 4-5 FFT validation of data

To determine an optimal cut-off value for the FFT, machine learning was applied to historical test data. Three optimisation algorithms, including genetic algorithm (GA), particle swarm optimisation (PSO), and artificial bee colony (ABC), were used to identify the most effective optimisation approach.

The objective functions for optimisation were expressed as:

$$\boldsymbol{Objective\ Function}: Minimize\left(abs\left(mean\left(Ballbar_{CW\,Data}\right) - mean\left(Accel_{CW\,Data}\right)\right) + abs\left(mean\left(Ballbar_{CCW\,Data}\right) - mean\left(Accel_{CCW\,Data}\right)\right)\right) \quad (4\text{-}3)$$

Equation 4-1 FFT optimal objective function

Using this objective function in Equation 4-1, the machine learning algorithms were initialised with the primary goal of identifying the optimal cut-off value in the fewest possible iterations.

This ensures rapid recalibration of the system in scenarios where the COBOT is repositioned or performs a different operation.

Table 4-1 Results of machine learning for optimal cut-off value

Search Approach	Iterations	Objective function	Selected cut-off value
	(Minimum is better)		
PSO (Selected)	**32**	**0.0679**	**0.265**
GA	61	0.0679	0.2655
ABC	100	0.068	0.2687

As shown in Table 4-1, the PSO method achieved the optimal cut-off value with the fewest number of iterations while maintaining performance comparable to the other algorithms. This automatic filter cut-off selection minimises the deviation between the known value and the extracted value from the raw data.

When applying the PSO-derived cut-off value to the FFT, the tracking system's error was compared against the ball bar results and found to have an error of 33.9 µm. This is a promising outcome, as the repeatability of the UR10e COBOT used in this study is 50 µm. indicating that the tracking system can, in theory, operate with greater precision than the COBOT's inherent repeatability.

The error results based on the PSO optimisation cannot be directly compared to previously collected data, as the accuracy of earlier results is uncertain due to variations in cut-off value settings, as shown in Table 4-2.

Table 4-2 Validation of PSO cut-off optimisation

COBOT motion	Ball Bar Results (mm)	Accelerometer results (mm)	Error (µm)
CW	100.0405	100.0104	30.103
CCW	100.0439	100.0061	37.776
Average Error			**33.9395**

The dynamic error measurement system was designed to operate seamlessly with most modern CNC machines, robotic arms, COBOTs, and machine controllers capable of exporting data to external devices. The system can function as a supervisory digital twin [88]. MATLAB was used to process and display the current XYZ position, while RoboDK provided a real-time visualisation of the system on a computer.

4.4 Evaluation experiment

4.4.1 Experimental setup and procedure

The ball bar test is widely recognised as a globally accepted standard for assessing machine accuracy, particularly in machining applications. Ball bars are measurement tools capable of providing error readings for both quasi-static and dynamic measurements. In this study, the ball bar test was used to evaluate the accuracy of the measurement approach, with ball bar data serving as the reference standard for comparison with superimposed position data obtained from the accelerometer measurement and the encoder of the COBOT.

A Renishaw QC20-W system was employed for this purpose, offering a measurement accuracy of 0.1 µm [89]. The setup of the ball bar test on the COBOT is illustrated in Figure 4-6. The

ball bar measured the linear distance between two endpoints using a transducer, enabling the radius from the centre to be determined as the ball bar moved. The measured data were used to assess the accuracy of the machine as well as its quasi-static and dynamic characteristics[61, 90].

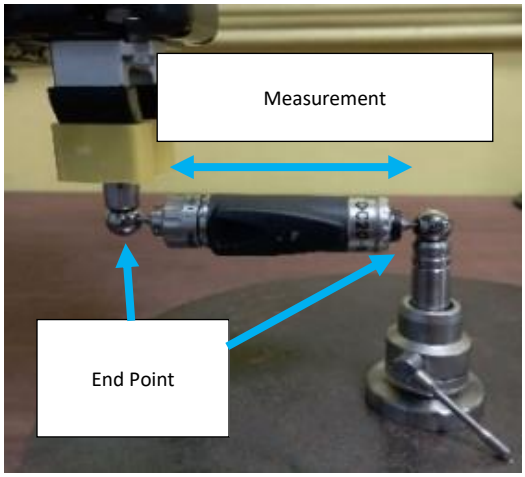


Figure 4-6 Set up of the ball bar test

The following assumptions were made in the ball bar test:

- The room temperature was assumed to be constant, as the test was performed in a very short span of time.
- The basement location of the test, with concrete floors, allowed the assumption that external vibrations did not influence the test.
- The ball bar readings were considered error-free; to ensure this, the ball bar was recalibrated between changes in feed rate.

The experiment was conducted in the following steps:

- A 100 mm radius ball bar test was conducted in the XY plane at different feed rates, ranging from 2000 mm/min to 6000 mm/min in increments of 1000 mm/min.
- Each feed rate test was repeated three times, and the results were averaged to improve reliability.

- Real-time positioning data from the COBOT controller and acceleration data were collected simultaneously.
- Acceleration data were filtered and double-integrated to compute displacement.
- The computed displacement was considered the dynamic error and superimposed onto the COBOT positioning data from the TCP.
- The superimposed data and the raw COBOT positioning data were then compared with the ball bar results to evaluate their relative accuracy with respect to the actual TCP position.

The limitations of the current experiment are:

- Although the COBOT supports six-axis motion, the current scope of dynamic error tracking is limited to three translational axes. Further adjustments are required to account for rotational movements of both the accelerometer and the COBOT.
- Ball bar tests were conducted only in the XY plane. The Z axis was not considered to reduce the number of experiments for each feed rate, as ball bars can measure only two axes at a time, requiring three separate tests for full 3D coverage.
- The current system sampling rate is limited to an average of 4.6 Hz due to constraints of the RoboDK and MATLAB APIs. Alternative communication methods are being investigated to increase the sampling rate.
- The accelerometers used have a bandwidth of 0.5 to 3000 Hz (±5%) and up to 5000 Hz (+10%), which may restrict the range of vibrations that can be accurately measured and processed by the system.

4.4.2 Processing results from Renishaw Software

To validate the tracking accuracy, data were collected using the program 'Renishaw Ball Bar 20.' This program was used both to set up the test and to acquire measurement data from the

ball bar. The test set-up page can be seen in Figure 4-7. When programming a ball bar test, a 1 mm in-feed motion is required to start the recording, and a 1 mm out-feed motion is required to end the recording. It is also considered a good practice to perform a circular lead-in and lead-out motion to ensure accurate measurement of the machine's circular movement.

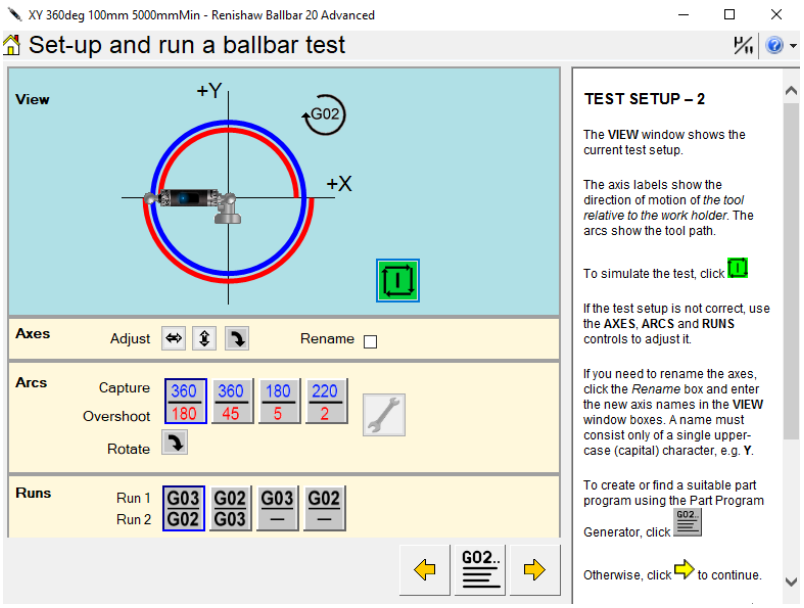


Figure 4-7 Renishaw Ball Bar 20 test set-up page

The ball bar test is conducted in clockwise (CW) and counterclockwise (CCW) directions. This can be seen in the G-code (see Figure 4-9) generated by the Renishaw ball bar 20, which was used to create the script code for the COBOT in RoboDK. The G code generated by Renishaw ball bar 20 for the COBOT can be seen in Appendix VI 16.1, also with the convert python script at Appendix VI 16.2.

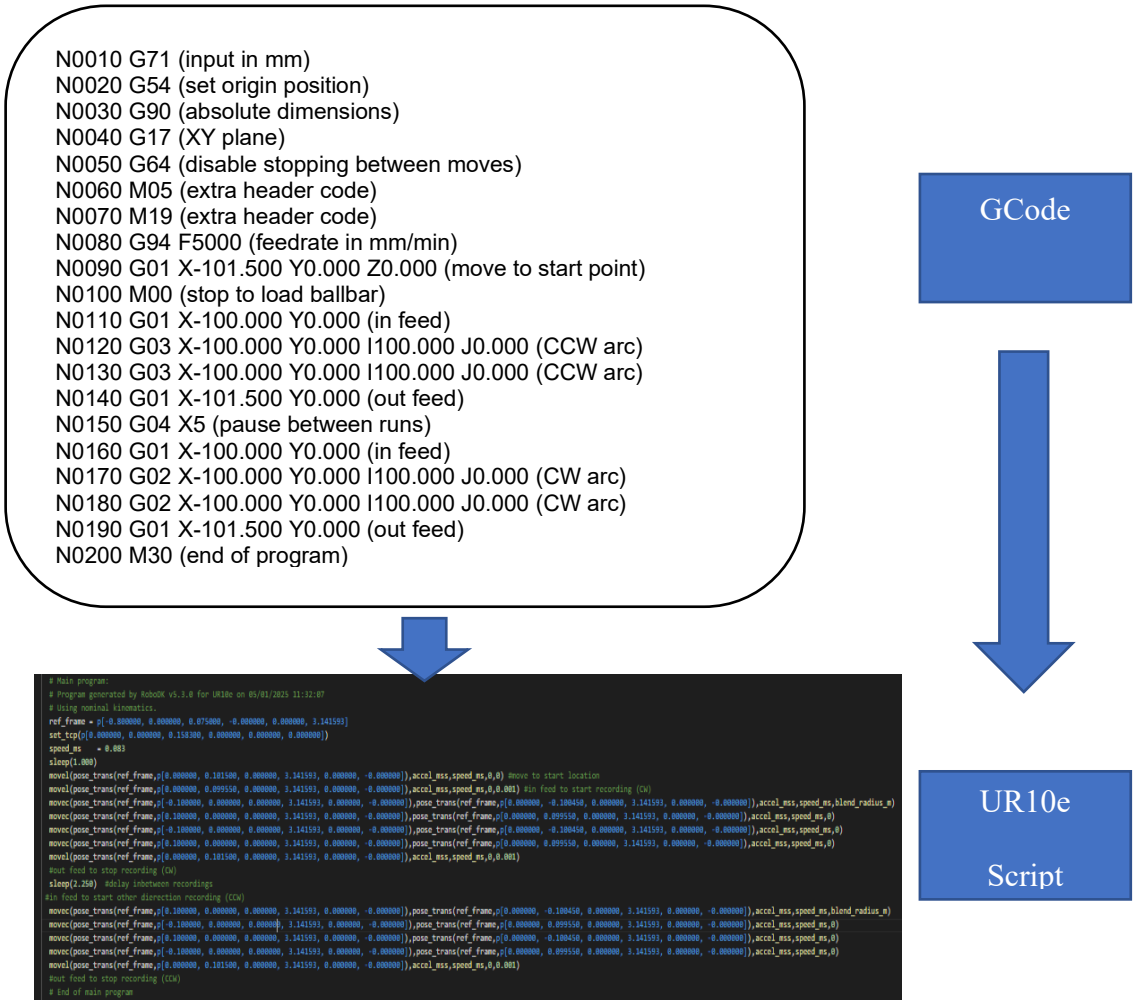


Figure 4-8 GCode Converted to UR10e Script

After the test is conducted, the results are saved and can be viewed in the Renishaw Ball Bar 20 software, where information about the machine's performance is displayed, as shown in Figure 4-9. The program does not directly provide raw data readings; instead, the data are saved as .b5r files, as illustrated in Figure 4-10.

The .b5r file format is associated with data used in specific scientific applications and can be opened in a text editor, where < and /> symbols separate the data using tags, as shown in Figure 4-10. Using this structure, the data can be extracted in MATLAB and organised into CW and CCW readings for further analysis. An example of a generated result from the ball bar readings can be seen in Appendix VI 16. 3.

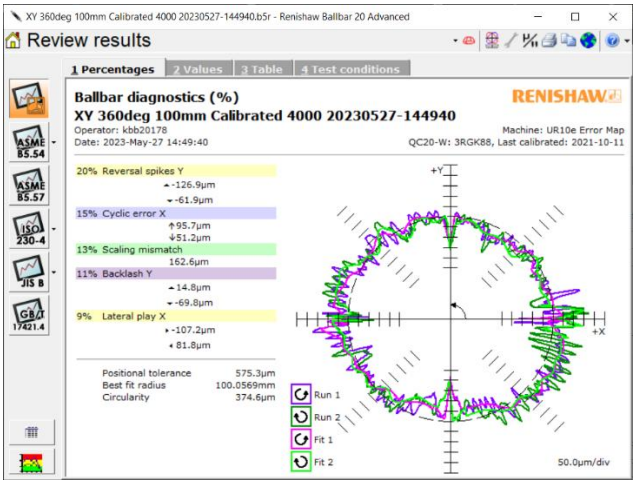


Figure 4-9 Renishaw Ball bar 20 results review

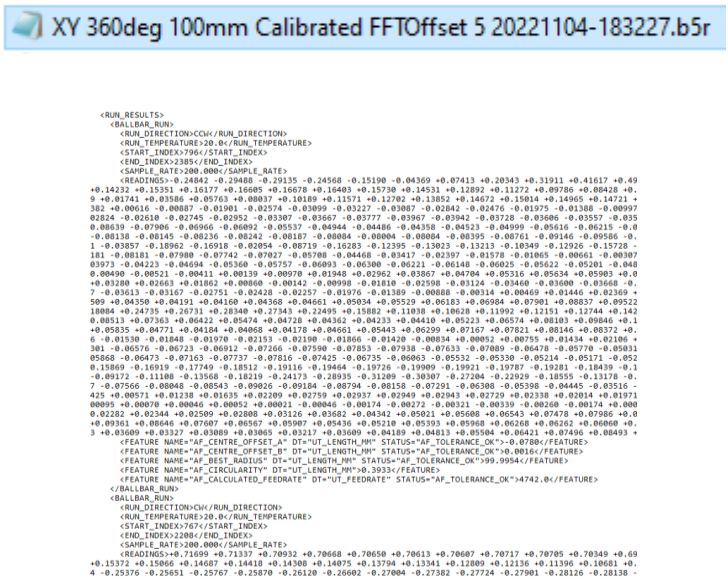


Figure 4-10 Renishaw Ball Bar text-based results

The readings are saved as data points representing deviations from the calibrated zero point of the ball bar. Positive values indicate an increased distance from the centre, while negative values indicate a reduced distance.

Before analysis, the lead-in and lead-out circular motions must be removed. This is done for both directions by using the START_INDEX and END_INDEX provided for the CW and CCW readings, as illustrated in Figure 4.11. This needs to be extracted for each test, as these values can vary slightly.

```
<BALLBAR_RUN>
  <RUN_DIRECTION>CCW</RUN_DIRECTION>
  <RUN_TEMPERATURE>20.0</RUN_TEMPERATURE>
  <START_INDEX>791</START_INDEX>
  <END_INDEX>2372</END_INDEX>
  <SAMPLE_RATE>200.000</SAMPLE_RATE>
  <READINGS>-0.15519 -0.23054 -0.24274 -0.
```

```
<BALLBAR_RUN>
  <RUN_DIRECTION>CW</RUN_DIRECTION>
  <RUN_TEMPERATURE>20.0</RUN_TEMPERATURE>
  <START_INDEX>805</START_INDEX>
  <END_INDEX>2412</END_INDEX>
  <SAMPLE_RATE>200.000</SAMPLE_RATE>
  <READINGS>-0.18173 -0.23103 -0.22383 -0.15
```

Figure 4-11 Start and end index values identified

This process allows the raw ball bar data to be collected; however, the data are recorded as deviations added to the test radius and do not provide XY coordinates of the COBOT's TCP. To enable a direct comparison between the COBOT's encoder-based and the accelerometer-based measurements, both encoder and accelerometer-based datasets need to be represented as radial distances from the centre.

This approach was chosen because the COBOT and accelerometer data are available in XY coordinates, and the ball bar data are only in radius, not in angle. Converting the radius to XY coordinates would require assumptions about the angle at each data point, potentially introducing errors.

The Pythagorean theorem is used to convert XY coordinates to a radial distance. At this stage, the angle is not required to evaluate whether the digital twin improves tracking accuracy.

Where a and b are the X and Y coordinates. This calculation is valid because the centre of rotation is consistent across all tests, so the only variables that change are the X and Y coordinates. This concept is illustrated in Figure 4-12.

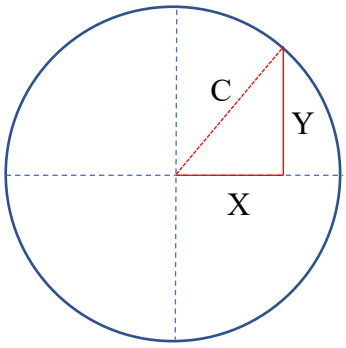


Figure 4-12 Radial error based of Pythagorean theorem

Using this approach, the COBOT encoder values and the measurement data can be quickly converted, ensuring that all datasets are in the same format and can be directly compared to the ball bar readings. The code to collect the data from the COBOT during the test can be seen in Appendix VI 16.4, and processing the ball bar readings and calculating the XY location of the data can be seen in Appendix VI 16.5.

4.5 Results and discussion

Figure 4-14 illustrates the comparison of the measurement results obtained by the ball bar, accelerometer and COBOT encoder. The accelerometer results are consistently closer to the ball bar measurements than the COBOT encoder, confirming the improved measurement accuracy achieved by this approach.

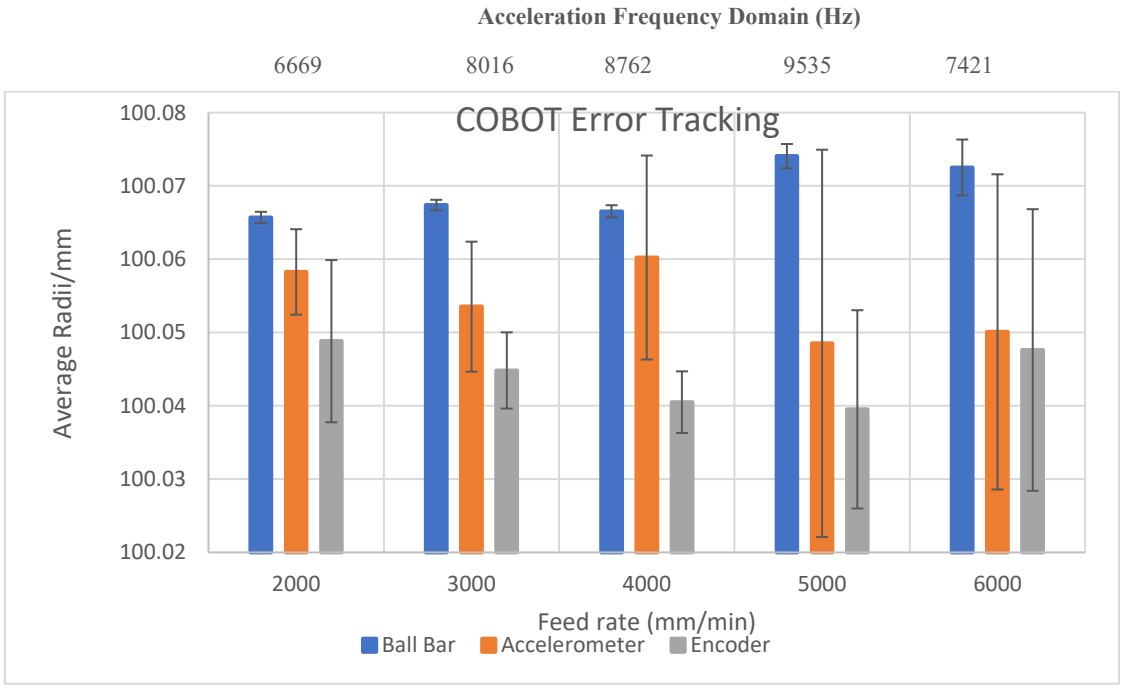


Figure 4-13 Comparison of COBOT error tracking results

The ball bar readings are considered the true TCP positions. The deviations between the ball bar results and the nominal values represent the actual total dynamic errors of the COBOT during the test. The results of the dynamic error obtained by the ball bar, accelerometer and

encoders are listed in Table 4-3. In the 100 mm radius ball bar test, the average dynamic error was 69.2 µm.

Table 4-3 Average radius of results for the COBOT double integration validation table

Dynamic Errors* in *100mm Radius Ball Bar Test					
Feed rate (*mm/min*)	**2000**	**3000**	**4000**	**5000**	**6000**
Ball bar (µm)	65.7	67.4	66.5	74.1	72.5
Accelerometer *(µm)*	58.3	53.5	60.2	48.5	50.1
Encoder (µm)	48.8	44.8	40.5	39.5	47.6
Difference between accelerometer and encoder data (µm)	9.5	8.7	19.7	9	2.5
Measurement error reductions (%)	56.2	38.5	75.8	26	10
Accelerometers frequency domain (Hz)	6669	8016	8762	9535	7421
Accelerometers mean frequency (Hz)	59.17	54.58	49.51	49.60	101.28

The accuracy of the accelerometer and COBOT encoder measurement results can be evaluated by comparing them with the ball bar results. The accelerometer measurement method achieved a significant improvement in measurement accuracy for dynamic errors, with measurement error reductions of up to 75.8% at 4000 mm/min. Across all feed rates, the average measurement accuracy of the proposed measurement method is 9.872 µm higher than the COBOT's built-in encoders, corresponding to a 41.3% reduction in measurement error. The results measured using the ball bar represent the total error, comprising the combined static and

dynamic errors of the COBOT during the test. At this stage, it is not possible to separate the individual contributions of these two error types. However, distinguishing between them is not necessary at this point, as the purpose of this test is to determine whether this approach can measure error with greater accuracy than the encoders alone. In later stages of the research, when corrective actions are applied, adjustments will be made only through modifications to the command instructions. Therefore, any observed reduction in error will correspond to a reduction in dynamic error, as no changes are made to the static error. From a frequency perspective, the frequency of measured deviations generally increases with the increase of feed rate, except at 6000 mm/min, as shown in Table 4-3. The mean frequency tends to decrease with the increase of feed rate until 6000 mm/min, where an out-of-trend behaviour is observed. This anomaly will be analysed in future studies.

The variation in measurement error by the proposed approach under different feed rates is likely due to the real-time communication capability of the data logger, which has an average sampling rate of 4.6 Hz. As the feed rate increases, fewer samples are collected, reducing the accuracy of the data. This limitation of slow sampling rate is currently attributed to constraints in the RoboDK and MATLAB APIs, and methods to overcome it are under investigation.

Despite this limitation, the advantages of the proposed dynamic error measurement approach are clear. Unlike the ball bar system, which is an offline, two-axis measurement tool, using accelerometers in the proposed measurement approach enables in-situ, real-time measurement of machine dynamic errors in three-dimensional space.

4.6 Summary

The chapter introduces a new measurement approach to obtain the dynamic error outside the servo loop using triaxial accelerometers. Two accelerometers are used in this approach, one at the COBOT wrist near the TCP and one on the base to enable. The subtraction of the

measurement results will remove background noise. The obtained acceleration signals are filtered in real time using FFT, and the optimal cut-off value is automatically selected using optimisation algorithms. Double integration of acceleration will obtain three-dimensional displacements at the TCP, which will be superimposed onto the COBOT's encoder data to obtain the total dynamic errors.

System operation and data acquisition, processing and visualisation are implemented in MATLAB and RoboDK. The validation of the proposed method is performed through a ball bar test under feed rates from 2000 to 6000 mm/min. The validation results show that the proposed dynamic error measurement approach is consistently closer to the ground truth ball bar data than encoder data. It achieves up to 75% improvements of measurement accuracy up to 75%. Key limitations of the proposed approach include the low effective sampling rate ~4.6 Hz imposed by RoboDK/MATLAB communication, reduced performance at higher feed rates due to fewer samples, and an out-of-trend frequency behaviour at 6000 mm/min that requires further investigation. Overall, the method demonstrates a transferable, in-situ approach to real-time 3D tracking of dynamic error.

This chapter has demonstrated that the novel method of tracking dynamic error using accelerometers is feasible for machine error monitoring, thereby further addressing Research Question 2. The results also indicate that this approach is valid for the COBOT in workpiece-handling applications, contributing towards answering Research Question 3.

This is evidenced by the improved tracking accuracy achieved when compared with the default encoder-based measurements, as validated against ball bar test results. The enhanced tracking capability enables the implementation of corrective actions during future movements based on the predicted errors.

Having demonstrated the effectiveness of this approach, it must now be tested on the hybrid milling machine to determine whether the method is applicable across multiple types of machines using this new dynamic error measurement technique. Further research presented later in this thesis will investigate methods for improving both the accuracy and the speed of the system through the application of machine learning techniques.

5 Chapter 5 Measurement of Dynamic Error of Hybrid Mill

5.1 Introduction

This chapter presents the measurement of dynamic errors of a hybrid mill by using the same approach. The hybrid mill used in this study is a custom-built 6-axis ultra-precision machine. It is equipped with four linear and two rotational ultra-precision aerostatic bearings, motional axes controlled by an Aerotech A3200 controller. The measurement and validation experiment will be focused on the X and Y machine axes.

5.2 Experimental setup

In this dynamic error measurement experiment, three triaxial accelerometers (PCB 356B18) will be mounted on the worktable (X, Y axes), spindle axis (Z axis), and the base of the hybrid milling mill. The locations of accelerometers are illustrated in Figure 5-1.

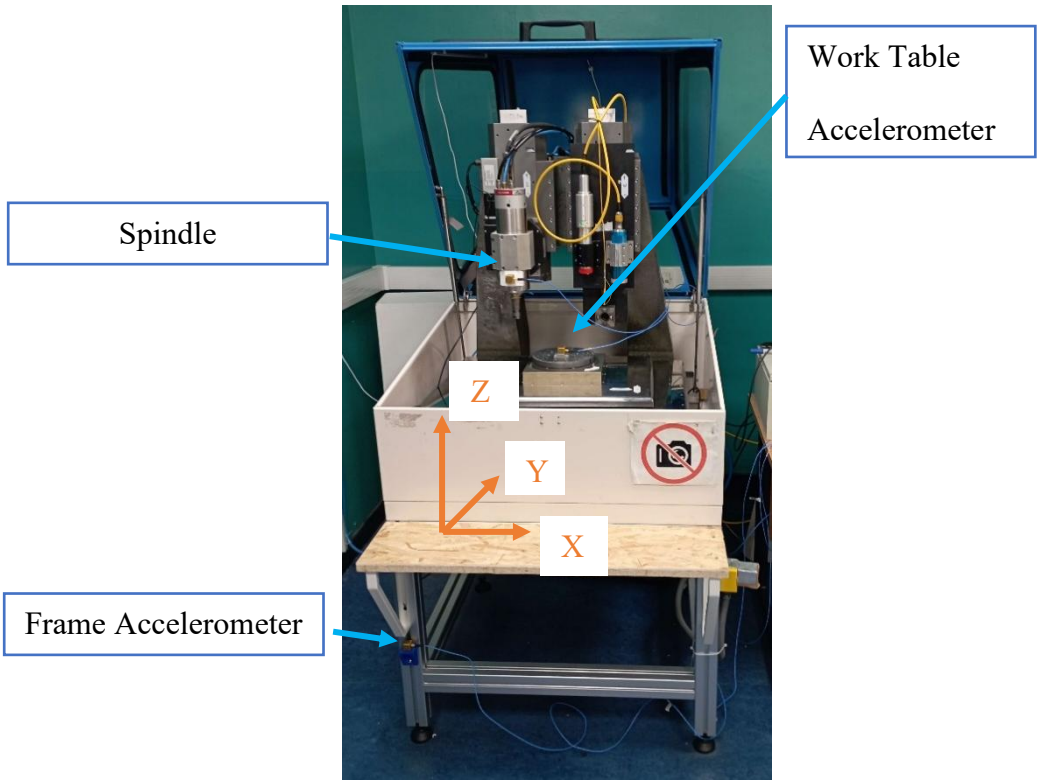


Figure 5-1 Hybrid Mill accelerometer locations

The three triaxial accelerometers provide acceleration data across nine axes, which will be continuously collected by the NI cDAQ-9174 card. When triggered d by a MATLAB programme, the most recent 1,500 samples for each axis are retrieved and stored for processing, including filtering and double integrations to obtain displacements. Adding the displacement measured by accelerometers to the machine encoder data, the total dynamic error will be obtained.

5.3 Measurement implementation

Data from the hybrid mill controller can be collected using MATLAB connected to the A3200 system, allowing access to all relevant data points from the controller. These data points are collected simultaneously and can be used in real-time for predictions. At the end of the process, the data are stored in a time-series matrix.

The specific data points collected via the A3200StatusGetItem function, along with their types and data structure, are summarised in Table 5-1.

Table 5-1 The hybrid mill available data points to be collected

Data point to collect	**Type of data collected**	**Structure of data**
Program Position Feedback (mm)	Knowing the current position of the mill axis relative to the set origin	Collection of time series positions for the XYZ axis measured by the encoders of the controller
Program Position Command (mm)	Knowing the intended position of the mill axis relative to the set origin	Collection of time series positions for the XYZ axis instructed by the controller

Program Position Error (mm)	Differences between the feedback and the command for each axis	Easy to compare the internal measured error of the machine
Current Feedback Average (Amp)	Knowing the current draw on each axis of the mill while in operation	Collection of time series current measurements that are internally averaged for the XYZ axis measured by the controller
Velocity feedback Average (mm/sec)	Knowing the measured velocity of each axis while in operation	Collection of time series velocity measurements that are internally averaged for the XYZ axis measured by the controller
Feed rate (mm/min)	Knowing what the intended programmed speed of the mill was at the time of motion	Collects the current programmed feed rate of the hybrid mill
Manual Feed Override (%)	Knowing if a modifier has been applied to the hybrid mill in its feed rate by the operator	Collects the percentage modification to the programmed feed rate manually set by the operator

Spindle Speed (RPM)	Knowing the spindle speed measured by the controller	Collects the feedback measurement of the spindle RPM

Communication between MATLAB and the A3200 is established using an API developed by Aerotech, with the code for setting up the communication and collecting data shown in Appendix VI 16.6. To initiate the connection, the code shown in Figure 5-2 must be added at the beginning of the MATLAB script, allowing the controller to be referenced using the variable "`handle`"

```
%%Aerotech set up
arch = computer('arch');
addpath('x64') %location of pre made functions
handle = A3200Connect;%varible to call the hybrid mill
```

Figure 5-2 AreoTech and MATLAB API set up.

Once the connection between MATLAB and the hybrid mill is established, data can be collected in real time using various commands. The primary command used for acquiring position data is "`A3200StatusGetItem`". Figure 5-3 illustrates an example of how the command position is collected; the same approach applies to collecting feedback position and position error.

This method of data acquisition from the controller is used to gather several key parameters, including command position, position error (feedback – command), average axis current draw, manual feed override (MFO), and spindle speed.

```
%pos command XYZ
    %A3200StatusGetItem(controllor, axis, data to collect)
    XProgramPositionFeedback = A3200StatusGetItem(handle, 0, A3200StatusItem.ProgramPositionCommand, 0);
    YProgramPositionFeedback = A3200StatusGetItem(handle, 1, A3200StatusItem.ProgramPositionCommand, 0);
    ZProgramPositionFeedback = A3200StatusGetItem(handle, 2, A3200StatusItem.ProgramPositionCommand, 0);
    ProPosCommand = [ProPosCommand;XProgramPositionFeedback,YProgramPositionFeedback,ZProgramPositionFeedback];
```

Figure 5-3 Aerotech get item command example.

Acceleration data are collected using the National Instruments (NI) API for MATLAB, which enables communication with the connected sensors. The connection is established by calling "`daq ('data logger name')`" in MATLAB. Once the connection is created, the sensors connected to the data logger can be configured, as illustrated in Figure 5-4.

```
accel = daq("ni");
    %S = spindel accel
    S0 = addinput(accel, "cDAQ1Mod1", 0, "Accelerometer");%X
    S1 = addinput(accel, "cDAQ1Mod1", 1, "Accelerometer");%Y
    S2 = addinput(accel, "cDAQ1Mod1", 2, "Accelerometer");%Z
    S0.Sensitivity = 1;
    S1.Sensitivity = 1;
    S2.Sensitivity = 1;
    %T = Table accel
    T0 = addinput(accel, "cDAQ1Mod2", 0, "Accelerometer");%X
    T1 = addinput(accel, "cDAQ1Mod2", 1, "Accelerometer");%Y
    T2 = addinput(accel, "cDAQ1Mod2", 2, "Accelerometer");%Z
    T0.Sensitivity = 1;
    T1.Sensitivity = 1;
    T2.Sensitivity = 1;
    %B = Base Accel
    B0 = addinput(accel, "cDAQ1Mod3", 0, "Accelerometer");%X
    B1 = addinput(accel, "cDAQ1Mod3", 1, "Accelerometer");%Y
    B2 = addinput(accel, "cDAQ1Mod3", 2, "Accelerometer");%Z
    B0.Sensitivity = 1;
    B1.Sensitivity = 1;
    B2.Sensitivity = 1;
```

Figure 5-4 National Instruments and MATLAB API set up.

The NI-9234 data acquisition card used in this study is designed for sound and vibration measurements. The input channels can be declared as accelerometers, which automatically apply the appropriate settings and provide acceleration values in g-forces (G). Later in the code, these values are converted to mm/s².

Acceleration data are collected continuously, streaming to the data logger. When requested, a specified number of scans are retrieved from the buffer. These data points are then processed using FFT filtering, converted to mm/s², and double-integrated to calculate displacement, which serves as the predicted dynamic error.

Additional data required for MATLAB to train and operate the hybrid mill dynamic error predictor includes acceleration measurements from the spindle, table, and frame. The same triaxial accelerometers (PCB model 356B18) and data acquisition system used for the COBOT are employed here. Acceleration signals are collected via a data logger (National Instruments cDAQ-9174) and transmitted to a data acquisition card (NI-9234).

5.4 Experimental validation

5.4.1 Validation experimental setup and procedure

The validation experimental setup, with all components labelled, is illustrated in Figure 5-5.

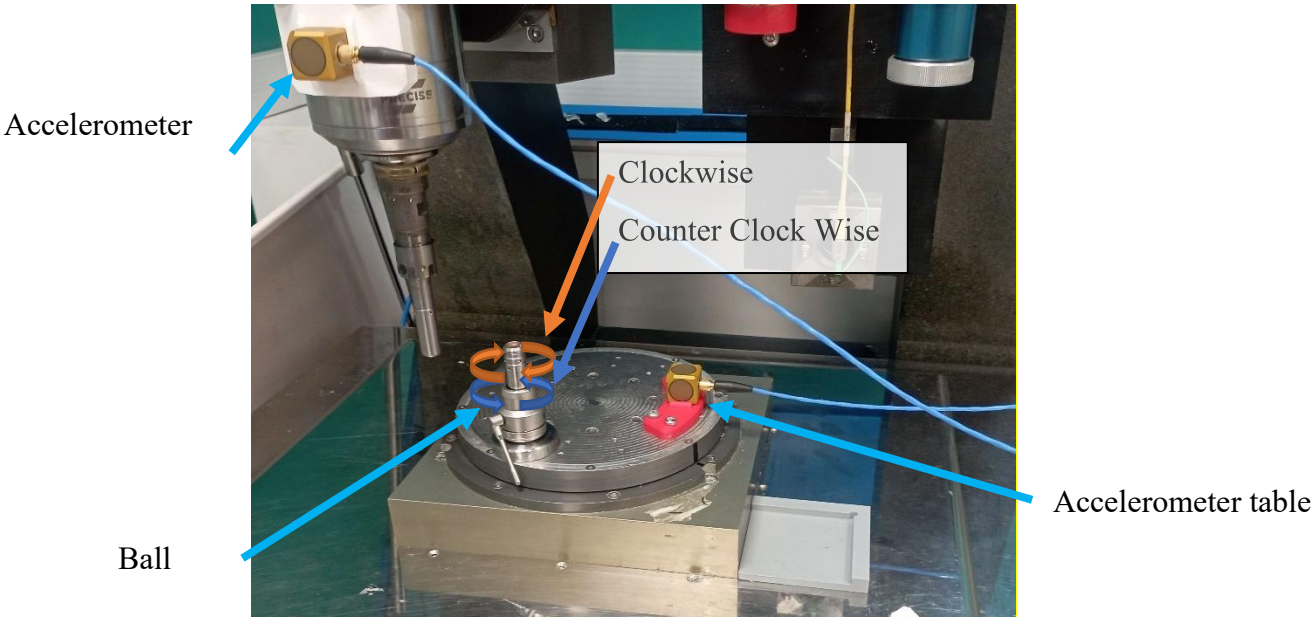


Figure 5-5 Hybrid mill ball bar set-up

The accelerometers are mounted on the hybrid mill using custom 3D-printed parts. The spindle mount is secured with four magnets, the table mount with four bolts, and the frame mount with two bolts. Each accelerometer is attached to its respective mount using a single 10/32 bolt. The designs of the mounts are illustrated in Appendix III. The three accelerometers attached to the hybrid mill are aligned with the mill axes. The table accelerometer is secured using four bolts and is aligned through adjustments to the C-axis rotation. The frame accelerometer is bolted onto a leg of the extruded aluminium frame, with guide walls incorporated into the mount to prevent rotational movement of the accelerometer. The final accelerometer, attached to the spindle, is secured using magnets to the spindle housing. It is aligned with the spindle mount and also uses guide walls to prevent rotational movement when secured to the hybrid mill. For validation, a ball bar test will be conducted using a Renishaw QC20-W system, limited to the XY plane. Due to the hybrid mill's small work envelope, a 50 mm radius circle will be used.

Data will be collected via MATLAB, which communicates with the Aerotech A3200 controller to retrieve hybrid mill feedback positions, command positions, positioning errors, and acceleration signals. During operation, MATLAB will calculate the dynamic error by combining the feedback position from the encoder with the measured displacement.

Encoder data and displacement measured by accelerometers will then be compared to the ball bar measurement results. As the QC20-W ball bar has an accuracy of 0.1 μm and operates externally to the control loop, it can measure true dynamic errors, providing a ground truth for validation of the proposed measurement approach.

To evaluate the accuracy and repeatability of the proposed measurement approach, the experiment will be conducted at multiple feed rates, with each feed rate tested three times. The selected feed rates are 500, 250, and 100 mm/min, respectively, which are commonly used in ultra-precision manufacturing. The complete set of planned tests is summarised in Table 5-2.

Table 5-2 Hybrid mill double integration validation test conditions

Feed rate (mm/min)	**Test 1**	**Test 2**	**Test 3**
100	1	2	3
250	1	2	3
500	1	2	3

The process of each test will be conducted in these steps:

- The ball bar is calibrated at the start of each set of feed rate tests.
- Initial positioning data and offset acceleration readings are collected from the hybrid mill.

- Acceleration data are corrected for offsets on each accelerometer and filtered using FFT.
- Acceleration readings from the table and spindle have the base accelerometer readings subtracted to remove the effects of external vibrations.
- Acceleration values are converted to mm/s^2.
- Double integration is performed on the filtered accelerations from the table and spindle to calculate displacement along each axis.
- Collected position data and calculated displacements are saved for further analysis.

After completing all tests, the data will be processed to extract three key datasets: ball bar positioning data, encoder positioning data, and displacement measured by accelerometers. Comparing these datasets will allow assessment of the effectiveness of the proposed method for tracking dynamic errors of the hybrid mill.

5.4.2 Data processing in validation experiments

The data to be processed are divided into two sets: one from the hybrid mill and one from the ball bar. The methods used to collect and process each dataset for the tracking system will be described, along with the procedure for combining the data to evaluate whether the digital twin provides more accurate hybrid mill tracking than the encoder readings alone.

The ball bar data are processed similarly to previous steps, starting from the generation of the ball bar path, followed by data collection, and calculation of XY coordinates based on the radius and calculated angle. An additional step is implemented to synchronise the hybrid mill and accelerometer data with the ball bar measurements.

This step uses a K-Nearest Neighbours (KNN) search to identify the closest individual data points, rather than treating the data as averaged values. KNN searches find the nearest

neighbour in X for each point in Y, allowing the nearest hybrid to mill to position to be matched automatically to each ball bar reading in both CW and CCW directions.

Once the ball bar and hybrid mill data points are aligned to their nearest neighbours, comparisons can be made between the true error (ball bar readings), the default error tracking (encoder readings), and the digital twin tracking (encoder data combined with the predicted dynamic displacement).

5.5 Results and discussion

Table 5-3 show the mean absolute dynamic errors measured by the ball bar test, encoders and accelerometers, as individual readings may be positive or negative. Across all tests, the average dynamic error measured by the ball bar is 6.72 µm.

Table 5-3 Hybrid mill double integration validation results

	Mean results (um)			**Tracking error (um)**		**Tracking error (%)**	
Feed rates	Ball bar	Encoder	Acc	Encoder	Acc	Encoder	Acc
500	6.372	0.193	3.802	6.179	2.570	97.0	40.3
250	6.097	0.041	2.031	6.056	4.066	99.3	66.7
100	7.692	0.013	0.267	7.679	7.425	99.8	96.5
Mean overall	6.720	0.082	2.033	6.638	4.687	98.7	67.9

The experimental results indicate that the proposed measurement approach improves the dynamic error measurement accuracy of the hybrid mill, producing readings that are closer to the ball bar measurements. This is shown in the results, with tracking error being reduced to

40.3%, but with the reduction in effectiveness at slower speeds, more investigation will have to be conducted to determine the reliability of this approach.

5.6 Summary

The chapter presented a measurement test of dynamic errors of a hybrid mill, and the evaluated test has demonstrated the effectiveness of the proposed approach. However, a new challenge arises due to limitations in communication rates and the computational load required to process the necessary data. In the current method, each data point requires many calculations: FFTs are applied to the X and Y axes of both the table and frame, followed by double integration of the combined acceleration values. While this approach can improve accuracy, as the volume of processed data increases due to higher sampling rates, the computational demands grow exponentially. This creates a limitation in further improving the method without drastically increasing computational load. Therefore, a more efficient method for calculating displacement from the available data is required.

Building on this, the potential to use machine learning to predict dynamic errors directly from raw acceleration signals will be explored in the following chapter. Predicting dynamic error using machine learning offers several advantages: it reduces computational load by eliminating the need for filtering and double integration, enables faster position calculations, and potentially improves overall tracking accuracy.

The results of this experiment validate the proposed error-tracking method for the hybrid milling machine, demonstrating that this approach is capable of tracking machine error more effectively than the encoder-based measurements alone. The use of accelerometers for dynamic error tracking has therefore been shown to function successfully on the hybrid mill, addressing Research Question 2 by providing a method capable of measuring errors occurring outside the machine control loop.

However, the effectiveness of the system is lower than that observed for the COBOT. The underlying cause of this difference will require further investigation in subsequent research. Nevertheless, these results provide a foundation for further work to implement corrective actions and to integrate machine learning techniques in order to improve both the accuracy and speed of the digital twin when predicting dynamic errors.

Considering the results obtained from both the COBOT and the hybrid mill, it can be concluded that Research Question 2 has been addressed, as an effective approach for measuring dynamic errors outside the control loop has been demonstrated across two different machine platforms using this novel method. These findings suggest that the approach may apply to other machines; however, additional research would be required to validate this across a wider range of systems, which lies beyond the scope of the present study.

6 Chapter 6 Predictive Digital Twin-driven Dynamic Error Compensation

6.1 Introduction

Building on the success of dynamic error tracking using accelerometers, this Chapter investigates the potential of predicting dynamic errors using machine learning (ML) and AI approaches, aiming to enhance both the accuracy and operational speed of the system.

While advanced AI approaches such as deep learning can produce accurate predictions, they often rely on 'black box' models that provide little transparency regarding how predictions are generated [91]. To address this limitation, this study employs a transformative AI approach that represents dynamic errors through explainable mathematical expressions.

Such a method, known as explainable AI (XAI), has gained popularity due to increasing demand for fair, unbiased, and transparent AI decision-making. The goal of XAI models is to provide insight into how predictions are derived, rather than functioning as opaque systems. By revealing the key features and data patterns influencing predictions, XAI enhances operator trust in the algorithm Deeks et al. [92]. This approach will be applied to investigate dynamic error prediction for both the COBOT and the hybrid mill.

6.2 Predictive digital twin-driven dynamic error control for COBOT

6.2.1 Method

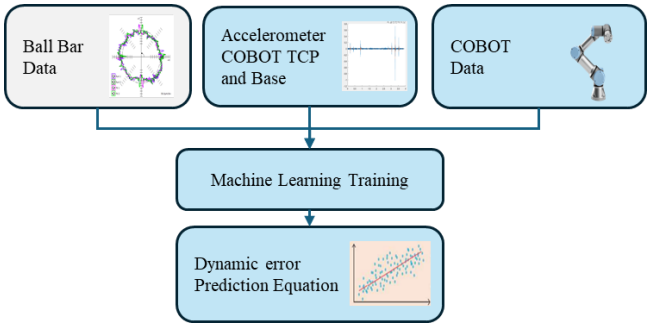
The application of XAI for COBOT dynamic error prediction represents a novel and largely unexplored research area. XAI offers a significant advantage by providing transparency and interpretability in the prediction process, enabling users to understand how the model arrives at its conclusions. This insight can lead to a deeper understanding of the underlying causes of

dynamic errors in COBOTs, potentially revealing issues that might be overlooked by traditional black-box models.

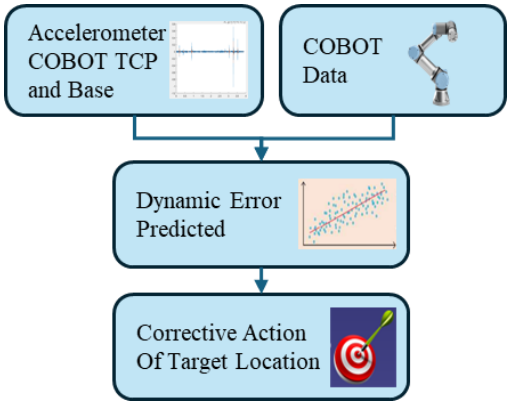
By leveraging XAI, this approach not only enhances the precision and reliability of COBOTs but also provides clear justifications for real-time corrective actions, improving trust and operational efficiency. Such transparency is particularly valuable in high-stakes environments, such as ultra-precision manufacturing, where understanding the rationale behind predictions or corrective actions can facilitate debugging, support system optimisation, and reinforce confidence in automated processes.

The structure of the COBOT digital twin is illustrated in Figure 6-1, where key components are represented as blocks and arrows indicate the direction of data flow. For training the system (Figure 6-1(a)), Acceleration data are collected from two accelerometers—one at the wrist near the TCP and the other at the base of the COBOT—along with COBOT encoder data and ball bar measurements. The ball bar data serve as the ground truth, allowing the XAI model to generate an equation that predicts the true position based on key features extracted from the accelerometer signals. These data are processed for machine learning to produce a predictive equation for the COBOT's dynamic error.

Once training is complete, a prediction equation is generated, and ball bar measurements are no longer required during operation (Figure 6-1(b)). The trained model then provides real-time corrections to COBOT motion, ensuring improved accuracy during automated tasks.



(a)



(b)

Figure 6-1 (a) Training and (b) operation structure and data flow of the proposed predictive digital twin

To support the XAI model in generating a predictive equation, the acceleration data are further processed to extract key features. This step ensures that the most relevant information is available for constructing a mathematical model capable of accurately predicting dynamic errors. The selected features capture a comprehensive set of characteristics from the acceleration signals that are critical for reliable dynamic error prediction. These features are:

- Mean Absolute Value (MAV): A statistical measure that quantifies the average magnitude of a signal, regardless of its direction (positive or negative). MAV is particularly useful for analysing signals such as vibrations, accelerations, or electrical currents, where the magnitude of the signal is more important than its sign.

- Variance: Measures the spread or dispersion of a signal's values around its mean. In signal analysis, variance provides insight into the fluctuation or energy of the signal over time.
- Root Mean Square (RMS): Quantifies the magnitude of a signal by representing its effective value or energy over a period. RMS is useful for oscillatory or varying signals, such as vibrations, electrical currents, and sound waves.
- Shape Factor: A dimensionless measure characterising the waveform of a signal. Calculated as the ratio of RMS to MAV, it provides insight into the distribution and shape of the signal waveform.
- Impact Factor: Evaluates the presence of sharp peaks or transients in a signal. Defined as the ratio of the maximum absolute value to the MAV, it highlights the impulsiveness or spikiness of a signal.
- Crest Factor: Quantifies the ratio of the peak amplitude to the RMS value of a signal. This dimensionless metric indicates how extreme the peaks of a signal are relative to its overall energy.
- Kurtosis: Measures the "tiredness" or sharpness of the peak of a signal's distribution. High kurtosis indicates sharp peaks and heavy tails, often associated with transient events or anomalies.
- Skewness: Quantifies the asymmetry of a signal's amplitude distribution around its mean, showing whether values tend to cluster more on one side of the mean than the other.
- Frequency: Represents the number of times a repeating event occurs per unit of time. In signals, it describes how often the signal oscillates or cycles and is a fundamental characteristic of time-varying signals.

- Band Power: Measures the total power of a signal within a specific frequency range, indicating the concentration of signal energy within that band. It is widely used for frequency-domain analysis of vibrations and other oscillatory signals.

6.2.2 Training data for dynamic error prediction

To train the XAI model, data must be collected that allows the input features from the accelerometers to be correlated with the outputs from the ball bar, enabling accurate predictions of dynamic error using raw acceleration data. For this purpose, a comprehensive training dataset was collected.

The training data were designed to capture errors within a representative movement volume of the COBOT. Ball bar measurements were used during data collection to ensure high accuracy, which limited the collection to two axes at a time and within a circular testing volume. Data was collected at three different radii, three different speeds, and across the XY, XZ, and YZ planes. Each test condition was repeated three times to provide the XAI model with sufficient information to learn the COBOT's dynamic behaviour and accurately predict dynamic errors during operation. The test conditions are summarised in Table 6-1.

Table 6-1 COBOT machine learning data collection test conditions

Radius (mm)	**Feed rates (mm/min)**			**Test Plans**		
100	2000	4000	6000	XY	XZ	YZ
150	2000	4000	6000	XY	XZ	YZ
300	2000	4000	6000	XY	XZ	YZ

The 150-radius test plans are illustrated in Figure 6-2. Figure 6-3 shows all the test plans and radii within the working volume for providing the training data.

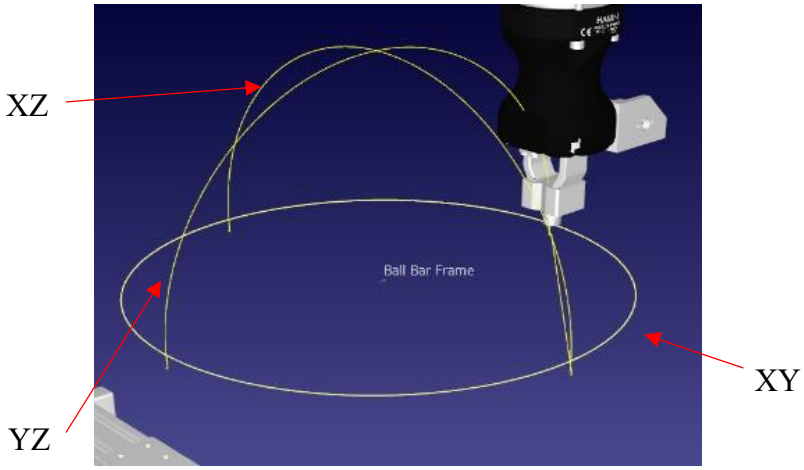


Figure 6-2 COBOT 150 test plans

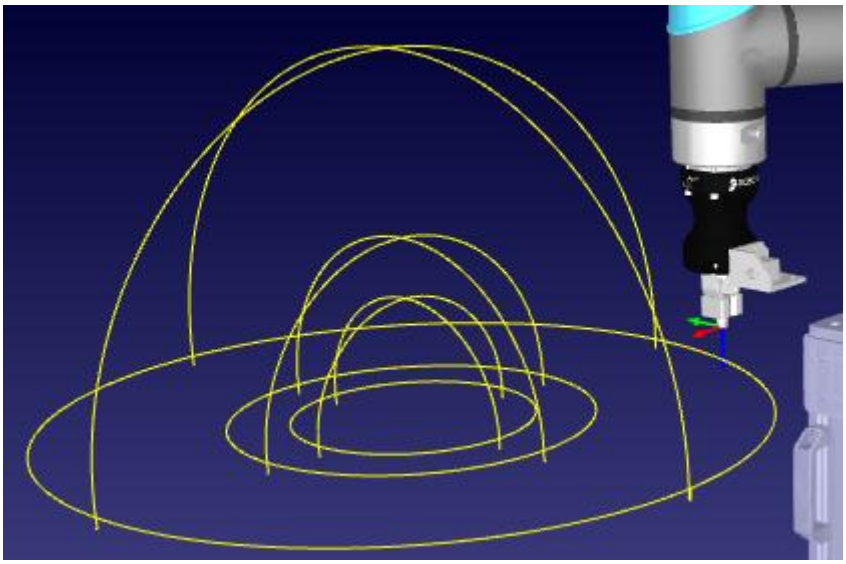


Figure 6-3 All COBOT test plans and radius

6.2.3 Processing of training data for dynamic error prediction

The processing of the training data followed a methodology similar to that used for the XY validation tests, but included data from the XZ and YZ planes and required formatting suitable for XAI model training.

To collect data for the XZ and YZ planes, two sets of test conditions were created for the COBOT, using the predefined start conditions. Each test was conducted at the relevant radius and feed rates, and repeated three times to ensure reliable data.

For the XZ and YZ tests, new motion paths were programmed to move the COBOT appropriately. For tests involving the Z plane, a 180° arc was executed, divided into 5° segments for motion alignment, while maintaining the same 1 mm feed-in and feed-out as in the XY tests. Motion was conducted in both CW and CCW directions. An example of this motion is shown in Figure 6-4, with the corresponding G-code generated from Renishaw Ball Bar 20 and the COBOT script used for execution shown in Figure 6-5.

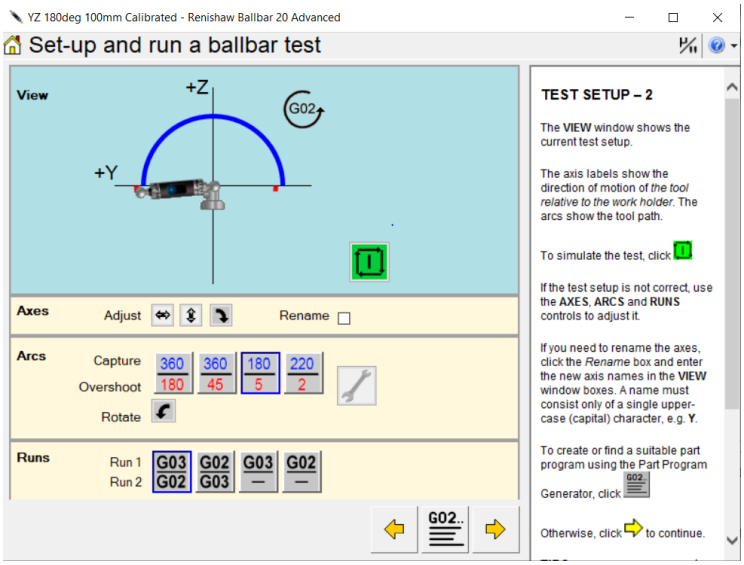


Figure 6-4 Renishaw Ball Bar 20 ZY set up.

N0010 G71 (input in mm)
N0020 G54 (set origin position)
N0030 G90 (absolute dimensions)
N0040 G19 (YZ plane)
N0050 G64 (disable stopping between moves)
N0060 M05 (extra header code)
N0070 M19 (extra header code)
N0080 G94 F6000 (feedrate in mm/min)
N0090 G01 X0.000 Y101.114 Z-8.846 (move to start point)
N0100 M00 (stop to load ballbar)
N0110 G01 Y99.619 Z-8.716 (in feed)
N0120 G03 Y-99.619 Z-8.716 J-99.619 K8.716 (CCW arc)
N0130 G01 Y-101.114 Z-8.846 (out feed)
N0140 G04 X5 (pause between runs)
N0150 G01 Y-99.619 Z-8.716 (in feed)
N0160 G02 Y99.619 Z-8.716 J99.619 K8.716 (CW arc)
N0170 G01 Y101.114 Z-8.846 (out feed)
N0180 M30 (end of program)

```
# Main program:
# Program generated by RoboDK v5.3.0 for UR10e on 06/01/2025 19:06:04
# Using nominal kinematics.
ref_frame = p[-0.800000, 0.000000, 0.075000, -0.000000, 0.000000, 3.141593]
set_tcp(p[0.000000, 0.000000, 0.158300, 0.000000, 0.000000, 0.000000])
speed_ms    = 0.083
sleep(1.000)
movel(pose_trans(ref_frame,p[0.000000, 0.101114, -0.008846, 3.141593, 0.000000, -0.000000]),accel_mss,speed_ms,0,0)
movel(pose_trans(ref_frame,p[0.000000, 0.100000, -0.008716, 3.141593, 0.000000, -0.000000]),accel_mss,speed_ms,0,0.000)
movec(pose_trans(ref_frame,p[0.000000, 0.000000, 0.100000, 3.141593, 0.000000, -0.000000]),pose_trans(ref_frame,p[0.000000, -0.100000, -0.008716, 3.141593, 0.000000, -0.000000]),accel_mss,speed_ms,blend_radius_m)
movel(pose_trans(ref_frame,p[0.000000, -0.101114, -0.008846, 3.141593, 0.000000, -0.000000]),accel_mss,speed_ms,0,0.000)
sleep(2.500)
# Skipping rounding for move with angle 180.0 deg
movel(pose_trans(ref_frame,p[0.000000, -0.100000, -0.008716, 3.141593, 0.000000, -0.000000]),accel_mss,speed_ms,0,0)
movec(pose_trans(ref_frame,p[0.000000, 0.000000, 0.100000, 3.141593, 0.000000, -0.000000]),pose_trans(ref_frame,p[0.000000, 0.100000, -0.008716, 3.141593, 0.000000, -0.000000]),accel_mss,speed_ms,blend_radius_m)
movel(pose_trans(ref_frame,p[0.000000, 0.101114, -0.008846, 3.141593, 0.000000, -0.000000]),accel_mss,speed_ms,0,0.000)
# End of main program
```

Figure 6-5 COBOT ZY GCode conversion to UR10e script

Once collected by the Renishaw Ball Bar 20, the Z-plane test data are presented in a format similar to that of the XY tests, as shown in Figure 6-6. This consistency allows the same method of data extraction from the saved .b5r files to be applied, using the START_INDEX and END_INDEX for the CW and CCW movements, as was done for the XY tests.

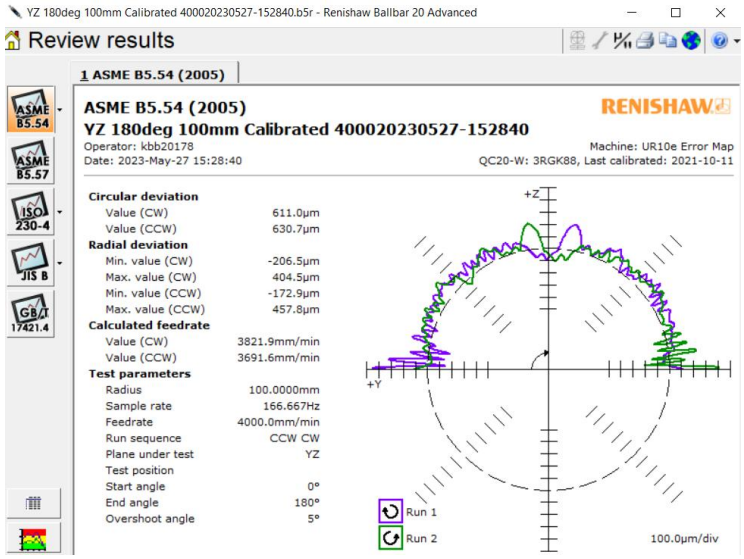


Figure 6-6 Renishaw Ball Bar 20 COBOT YZ results

After completing all 81 training tests and extracting the 10 key accelerometer features for each test, the collected data—including encoder values, acceleration features, and ball bar measurements—were compiled into a single dataset. This dataset was stored as an Excel document, with input data (encoders, accelerometer features, and test plan parameters) organised into separate columns, and output data (ball bar results) placed in a single column.

This structure allows the QLattice XAI algorithm to correlate the input features with the output measurements and generate a mathematical equation representing the relationship between all collected inputs and the ball bar output. Once an equation is generated, and if the R^2 value indicates a strong fit, testing of the predictive equation can begin. If the training data accurately capture the dynamic behaviour of the COBOT in motion, the XAI-generated equation should be capable of predicting dynamic error in real time based on unfiltered acceleration data.

6.2.4 XAI dynamic error prediction model

As previously outlined, the machine learning approach employed for dynamic error prediction in COBOTs utilises XAI, which generates interpretable equations providing clear insights into the prediction process. This section summarises the workings of the machine learning model and its training process, with detailed methodology available in [93] and [94].

The XAI method used in this study is QLattice, a symbolic regression tool that excels at discovering mathematical expressions accurately mapping input data to output results [95].

The QLattice algorithm was selected as it identifies mathematical models that describe the relationship between input variables and outputs in the form of interpretable mathematical expressions. This approach enables the identification of a minimal set of features required to produce an accurate predictive model. As the resulting model can often be expressed as a single equation, predictions can be calculated directly rather than requiring the execution of a full machine learning model. This reduces computational requirements and minimises latency between data collection and prediction. The QLattice determines an appropriate equation by composing mathematical operators and functions using the selected input variables and output target. The algorithm then iteratively refines these candidate models using reinforcement and evolutionary processes to improve their fit to the dataset. If no significant improvement in performance is achieved, the algorithm favours simpler model structures over more complex alternatives, improving interpretability and human understanding. As a result of the stochastic nature of the training process, a different mathematical model may be produced during each training iteration. Unlike conventional methods that rely on graph networks or genetic programming, QLattice models all potential input-to-output expressions as spatial path sets, allowing efficient exploration of expressions that minimise loss functions, such as Root Mean Square Error (RMSE), Wilstrup and Cave et al.[96].

Model performance is evaluated using a validation dataset that was not used during training. The dataset was divided into training and validation subsets using a split of [insert split]. The training process was repeated multiple times, with promising mathematical models retained for comparison. The best-performing model was then selected for use in the subsequent stages of the research. This enables users to understand how predictions are derived from the data,

evaluate the implications of hypotheses, and gain insight into the factors driving dynamic errors Brolos et al. [97].

For training QLattice, the input data consisted of acceleration signals, encoder positions, command data, and feed rate, while the output data was Ball bar positioning, serving as the ground truth for dynamic error assessment.

To help provide a clear comparison between QLattice and other regression models, the advantages and disadvantages are provided below.

QLattice

Advantage	Disadvantages
• Automatically identifies mathematical relationships between input variables and outputs. • Identifies the smallest set of features required to achieve accurate predictions. • Capable of modelling non-linear relationships between inputs and outputs. • Once trained, the resulting equation can often be evaluated directly, making prediction less computationally intensive than running a full machine learning model.	• The trained model may differ between training runs due to the stochastic nature of the algorithm. • Model training can take longer because the algorithm searches across many possible model structures. • Complex equations may be generated if appropriate complexity constraints are not applied. • The method is less widely used than traditional approaches, resulting in fewer established best practices for implementation and training.

Regression Models

Advantage	Disadvantages
• Training methods are well understood and widely implemented in statistical and machine learning software. • Statistical concepts such as confidence intervals, hypothesis testing, and error estimation are well established. • Regression models typically produce consistent results when trained multiple times on the same dataset.	• Limited ability to model complex non-linear relationships without additional feature engineering. • The model structure must be predefined, which can restrict the discovery of novel relationships within the data.

In summary, QLattice offers a powerful and transparent machine learning approach, generating clear mathematical formulas that enhance understanding and improve accuracy in dynamic error prediction for COBOT systems.

The training and testing phases were conducted using standard ball-bar circular motions with variable feed rates and radii. By covering a range of radii across a spherical 3D workspace, the model can predict and compensate for dynamic errors for any motion path within the COBOT's operational 3D space.

Using the data collected to train the XAI dynamic error prediction model is expressed as:

$$Radial\ Error\ = (0.0158 * F1 + 21.8) * (F2 - 0.362 \exp(-14200(0.121 - F3)^2 - 16.5(0.00161 * F4 - 1)^2) - 0.0209 \quad (6\text{-}1)$$

where F1, F2, F3 and F4 are Peak Amplitude (X-axis), Band Power (Y-axis), F3 Root Mean Square (X-axis), and F4 = Kurtosis (X-axis), respectively. They are acceleration features derived from data collected during ball bar tests across the XY, XZ, and YZ planes at various feed rates.

The error prediction equation can be expressed in the following form:

*Radial Error = (Scaling Term) * (Dynamic Behaviour Term)*

Thus, the predicted radial error consists of two main components: a scaling term and a dynamic behaviour term.

$$Scaling\ Term\ =\ (0.0158 * F1 + 21.8)$$

The scaling term acts as a linear scaling factor based on the peak acceleration amplitude of the X-axis (F_1). An increase in the peak acceleration amplitude, therefore, correlates with an increase in the predicted machining error. The constant value of 21.8 represents a baseline vibration-related contribution to the predicted error.

This relationship is consistent with the physical structure of the machine, as the X-axis supports the Y-axis. Any sudden disturbances or vibrations along the X-axis are therefore likely to propagate through the structure and influence the positional accuracy of the machine.

The dynamic behaviour term models how vibration energy and signal characteristics influence the error behaviour of the COBOT:

$$Dynamic\ Behaviour\ Term =\ F2 - 0.362 \exp\left(-14200(0.121 - F3)^2 - 16.5(0.00161 * F4 - 1)^2\right) - 0.0209 \quad (6\text{-}1)$$

The first term, band power of the Y-axis (F_2), represents the vibration energy within a specified frequency band. An increase in band power indicates greater vibration energy, which reflects

stronger dynamic excitation of the machine structure and therefore contributes to increased positional error.

The second component is a Gaussian-shaped correction factor that depends on the root mean square (RMS) acceleration of the X-axis (F_3) and the kurtosis of the X-axis signal (F_4). When the RMS value approaches 0.121, the exponential term increases significantly, indicating that deviations from this value strongly influence the predicted error. This suggests that the model has identified a critical vibration level associated with machine behaviour.

The third component relates to kurtosis, which measures the impulsiveness of the vibration signal. When the term $0.00161 \approx 1$ approaches unity, the contribution of the exponential term increases. This indicates that certain impulsive vibration characteristics are correlated with changes in the radial error.

The final component is a constant offset term, represented by 0.0209 (6-1), which acts as a correction factor derived from the training data and shifts the predicted value slightly downward.

Overall, the equation indicates that radial error is not purely linearly related to vibration magnitude. Instead, the presence of Gaussian terms suggests that specific vibration conditions are associated with increased machine error, rather than a simple proportional relationship between vibration amplitude and positioning error.

This supports the hypothesis that complex dynamic behaviour within the machine can be modelled using a data-driven digital twin. Such behaviour may not be captured using conventional machine controllers alone but can be identified through machine learning models that analyse acceleration features derived from vibration data.

The XAI model achieved a high training accuracy of 99.4%, as shown in Figure 6-. This high accuracy ensures that during real-world operation, the digital twin can effectively predict and compensate for dynamic errors, improving the overall positioning accuracy of the COBOT.

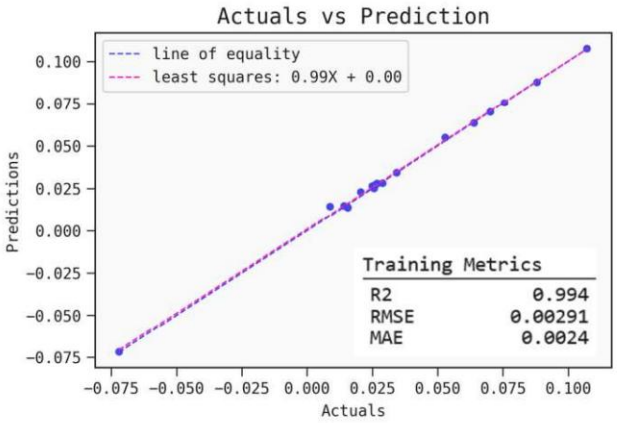


Figure 6-7 XAL model prediction accuracy

6.2.5 Comparison of QLattice with other ML algorithms

During initial development, the effectiveness of QLattice was compared with several other machine learning algorithms, including both explainable and black-box models, to determine the most suitable approach for this task. The explainable algorithms tested included KNN and Decision Trees (DTree), while black-box models included Artificial Neural Networks (ANN), Random Forest Regression (RFR), and Extreme Gradient Boosting (XGB). All models were trained on the same dataset and evaluated using the R^2 value to measure how well each model explained the variance in the data, with results illustrated in Figure 6-8.

The QLattice model achieved the highest R^2 value among all tested models, indicating superior predictive accuracy. Unlike KNN and DTree, which showed lower performance, QLattice generates predictions using a known mathematical equation, making it both accurate and computationally efficient.

While the black-box models (ANN, RFR, XGB) demonstrated strong predictive capabilities, they lacked interpretability compared to QLattice. The combination of high R^2 performance

and computational efficiency underscores QLattice's suitability for real-time dynamic error prediction in COBOT systems, making it the most accurate and practical choice among the models evaluated.

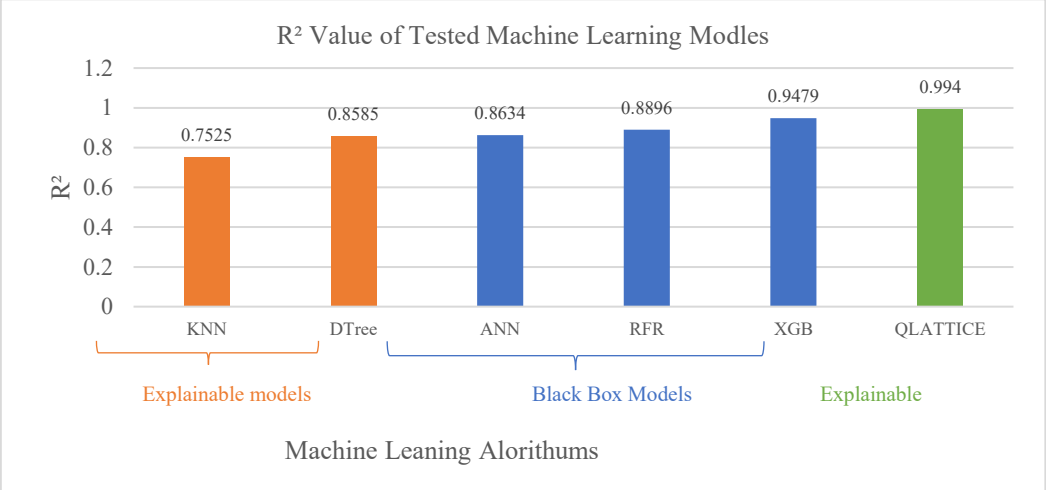


Figure 6-8 Comparison of different machine learning algorithms

This experiment was conducted to evaluate the effectiveness of the digital twin in reducing the dynamic error of a COBOT during real-time operations. The XAI model was trained to minimise the error between the ground truth, obtained from ball bar tests, and the encoder readings using the generated mathematical equation.

6.2.6 Dynamic error compensation approach

A dynamic error compensation approach was developed based on the XAI dynamic error prediction model shown in Equation 6-1. The corrective action concept involves shifting future targets based on the predicted error, so that the COBOT's motion compensates for expected deviations. To test this approach, a simple experiment was performed in which the COBOT moved along a semi-circular path. The experiments were conducted at feed rates of 2000, 4000, and 6000 mm/min, with a radius of 150 mm, and were repeated three times both with and without compensation.

Due to limitations in the available equipment and data collection methods, the programmed path for the COBOT was constrained to a short circular arc. Despite this, the experiment was sufficient to evaluate the feasibility of applying corrective actions in real time.

A flow chart of the experiment and the path followed by the COBOT is shown in Figure 6-9. In this setup, the RoboDK control program generates the TCP path at the start of a programmed movement. By continuously updating future target positions based on the predicted dynamic error while the motion is being initiated, the corrective action allows the TCP targets to hover around the programmed locations, effectively reducing deviations from the desired trajectory.

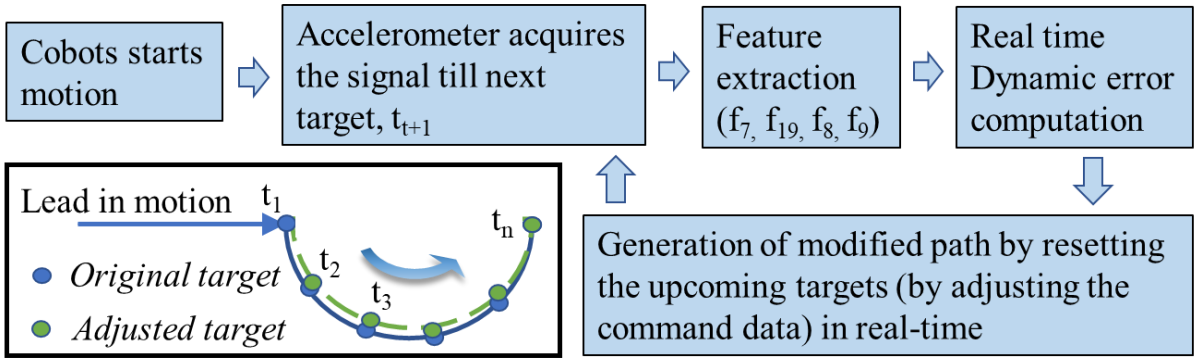


Figure 6-9 COBOT real-time corrective action approach

Figure 6-10 illustrates how the XAI-based predictions are integrated into the control loop. The XAI model generates real-time dynamic error predictions, which are used to adjust the target positions before they are executed by the machine, effectively compensating for deviations and improving overall tracking accuracy.

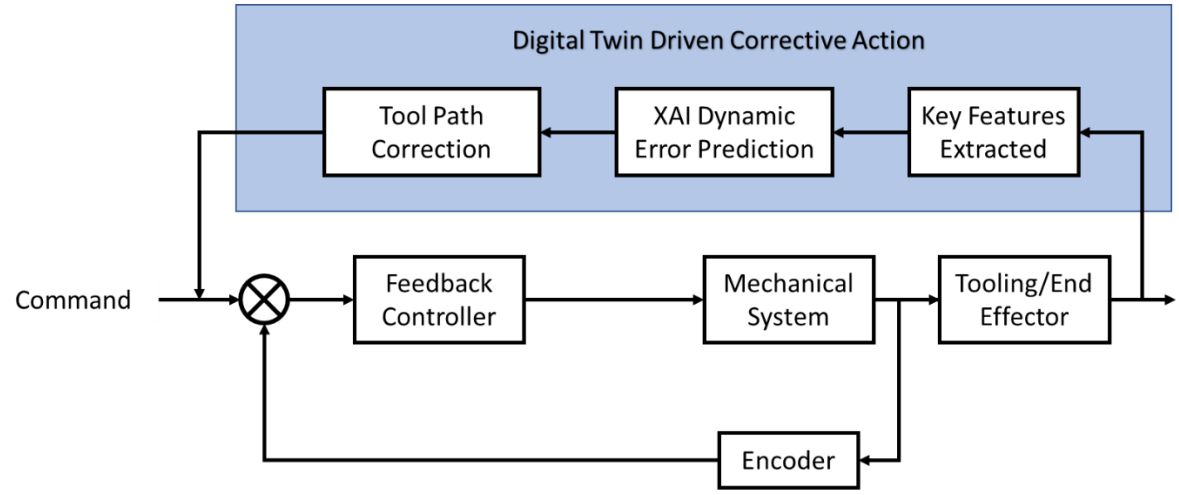


Figure 6-10 Dynamic error compensation based on XAI prediction model

Figure 6-7 shows the results of the evaluation experiment when using and not using the proposed dynamic error compensation method. The dynamic error of the COBOT motion has been reduced across the tested arc when the proposed dynamic error compensation approach is used by adjusting the programmed trajectory. The effectiveness of the proposed approach has been validated and demonstrated for real-time dynamic error compensation.

While promising, these results highlight the need for further in-depth testing, including experiments covering complete motion paths, to fully evaluate the effectiveness of the corrective action methodology.

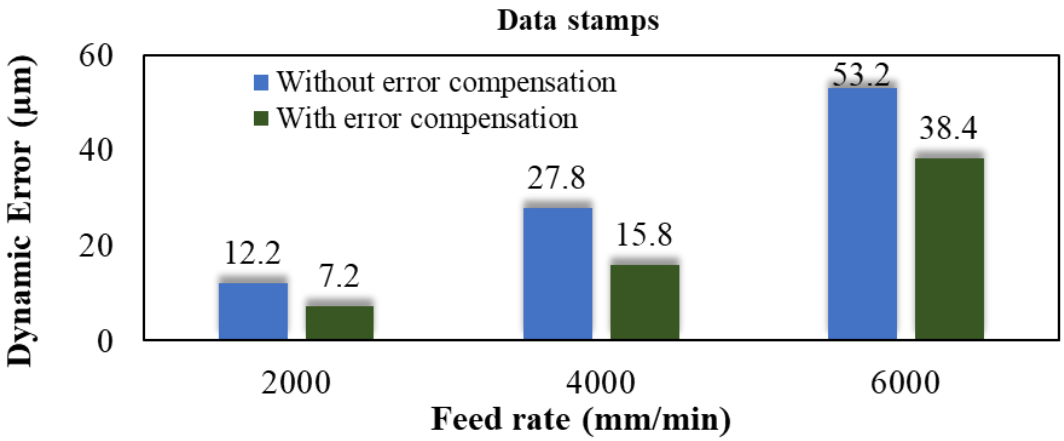


Figure 6-7 COBOT Real-time corrective action validation results

It is important to note that the current validation of the XAI-based corrective action was conducted only in the XY plane, meaning the system was tested in two dimensions. While these initial results demonstrate the practicality of the approach compared to the double integration method, further testing is required to assess its effectiveness in three dimensions. Future work will expand this approach to full 3D motion to determine whether the digital twin can reliably reduce the dynamic errors of the COBOT for real-world applications.

The effectiveness of this corrective approach is strongly influenced by the rate at which the system can collect relevant data, extract the necessary features, predict the dynamic error, and apply the corrective action to the COBOT or hybrid mill. System latency plays a critical role in determining the achievable reduction in dynamic error, as excessive delay may result in the predicted corrective action no longer being relevant to the machine's current motion.

The requirement for a low-latency system represents one of the advantages of the explainable artificial intelligence (XAI) approach used in this research. The prediction equation generated by the QLattice can be evaluated directly, which is computationally less demanding than passing all extracted features through a trained regression model. This enables faster prediction and supports the real-time application of corrective actions.

Testing and validation results indicate that the primary source of latency arises from communication between the different software programs used within the system. This latency will therefore be monitored throughout the implementation of the framework. Any future improvements that reduce communication delays between system components would likely improve the effectiveness of the corrective control and represent a potential area for further research.

The structure of the digital twin for the hybrid mill will differ from the COBOT one due to the control system differences, programming of tool pathing, data collection and corrective action.

From this, a new structure has been developed to allow for the changes to the machine. The design of this digital twin has been designed to be as universal as possible.

6.3 Predictive dynamic error compensation for the hybrid mill

6.3.1 Dynamic error test experiment on a hybrid mill for training a machine learning model

Although the controller for the COBOT and the hybrid mill differ, the exact implementation of predictive dynamic error compensation action cannot be identical; however, the underlying dynamic error prediction and compensation concept remains the same. To generate the training data for the machine learning prediction model, a ball bar test experiment will be conducted to characterise the dynamic behaviour of the hybrid mill.

Ball bar measurements data will be collected for the XY plane at a radius of 50 mm. The X and Y axes are prioritised for this initial training because the Z-axis of the machine was designed to exhibit lower dynamic errors than the XY axes, which may compromise the effectiveness and cost of applying dynamic error compensation in the Z-axis. The 50 mm radius is chosen as it represents the largest feasible circle within the hybrid mill's work envelope for ball bar testing.

The experiment will be conducted at feed rates of 500, 250, and 100 mm/min, respectively, with each condition repeated three times. A fourth set of data will be reserved for testing the machine learning model. New data is collected to remove possible variation between testing conducted in chapter 5 and this data set for the 4th data set from testing. The complete test plan is summarised in Table 6-2.

Table 6-2 Hybrid mill machine learning training data test conditions

Feed rate (mm/min)	Test 1	Test 2	Test 3	Training Test data
100	1	2	3	4
250	1	2	3	4
500	1	2	3	4

The remainder of the experimental setup will follow the same procedures used for the previous dynamic error test experiment in Chapter 5, with a few key differences. Unlike the COBOT experiment, the acceleration data in this case will be processed directly without FFT filtering. The results demonstrated that filtering is unnecessary when using machine learning, and skipping this step reduces computational load and increases the effective sampling rate.

The results from the ball bar tests will be collected and processed using the same methods as in Chapter 5. In addition, a kNN search will be applied to identify the nearest ball bar reading for each mill position. This allows for a precise comparison between the hybrid mill's feedback position—or the digital twin corrected position—and the corresponding ball bar reading, enabling an accurate assessment of the machine's real positioning errors.

6.3.2 Hybrid mill machine learning model training procedure

With the data collected from the accelerometers, the hybrid mill, and the ball bar, it is necessary to combine these into a usable matrix for a machine learning programme to process and generate solutions for each axis. For optimal training, separate matrices will be created for the X and Y axes, with one model trained per axis. This approach allows each model to account for the unique dynamic characteristics of its respective axis, leveraging the hybrid mill feedback, accelerometer readings, and ball bar measurements.

The ideal training data for a machine learning method is structured as a time-series matrix, with each row representing a sample point and each column representing a specific feature. The final column contains the ground-truth dynamic error as measured by the ball bar as the output for the machine learning model training. To achieve this, all data sources must be synchronised and aligned correctly. This is done by creating a matrix of the data collected by the hybrid mill, resulting in columns of data of Feedback position (XY), Command Position (XY), Position Error (XY), Acis Current Average (XY), Axis Velocity Average (XY), Feed rate, MFO and spindle RPM.

Next, the key features from the acceleration data must be extracted from the 1,500 samples collected at each time step for the hybrid mill. To do this, the raw acceleration data is reshaped into a matrix with 1,500 rows per sample, where each row represents a single sample at the corresponding time stamp of the hybrid mill data. Feature extraction is then applied to each row, calculating the 10 key features (Mean Absolute Value, Variance, RMS, Shape Factor, Impact Factor, Crest Factor, Kurtosis, Skewness, Frequency, and Band Power) for each axis of the accelerometer. This process synchronises the hybrid mill data and the acceleration key features as they are collected at the same time.

These correspond to periods at the start during the feed-in, at the midpoint when the direction changes between feed-in and feed-out, and at the end during feed-out. These points are not valid for training because when the mill is stationary, no dynamic error occurs. Static points are identified by monitoring the X-axis command position. Any command position less than (Xposition−50) corresponds to the mill either moving to the feed-in/out position or being stationary, and such points are removed.

After removing static and feed-in/out points, the corresponding ball bar error measurements must be collected for the remaining moving data. The ball bar data is processed the same as the

COBOT tracking, where readings are extracted from the saved `.b5r` files using the defined start and end indices. Once valid readings are extracted, the XY position of each point is estimated by dividing the total number of readings by 360° to calculate the angle between points, and trigonometry is used to determine the XY coordinates.

To match the ball bar readings with the hybrid mill data, a kNN search is used. This ensures that for each feedback position in X and Y (both CW and CCW), the closest corresponding ball bar reading is selected, providing a one-to-one mapping of error measurements to mill positions.

With this mapping complete, the error for each hybrid mill data point along the X and Y axes is known. These errors are combined with the corresponding encoder readings and accelerometer key features across all feed rates to create the final training matrices for the XAI and RL models. This results in the training data being a matrix of all the hybrid mill data, acceleration key features and lastly the ball bar reading for each feed rate.

The final training matrix is constructed by combining the data from the three repeated ball bar tests at each feed rate, sequentially starting with 500 mm/min, followed by 250 mm/min, and ending with 100 mm/min. This results in a single matrix containing all nine experimental runs. Combining the datasets in this manner allows the machine learning algorithms to observe dynamic behaviour across a range of feed rates, improving their understanding of how the machine's dynamic signatures change with speed.

Once the training matrix is prepared, the training test data must be processed similarly. The test data is structured with the same columns for input features and rows representing time-series data points, but it only contains the new signal test data. This results in a separate test data matrix for each feed rate and for each axis, yielding six matrices in total. Keeping the test

data separate ensures that it is unseen by the machine learning algorithms during training, providing an unbiased validation of the model's predictive capability.

An illustration of the eight matrices and how they are used in training and validation is shown in Figure 6-8.

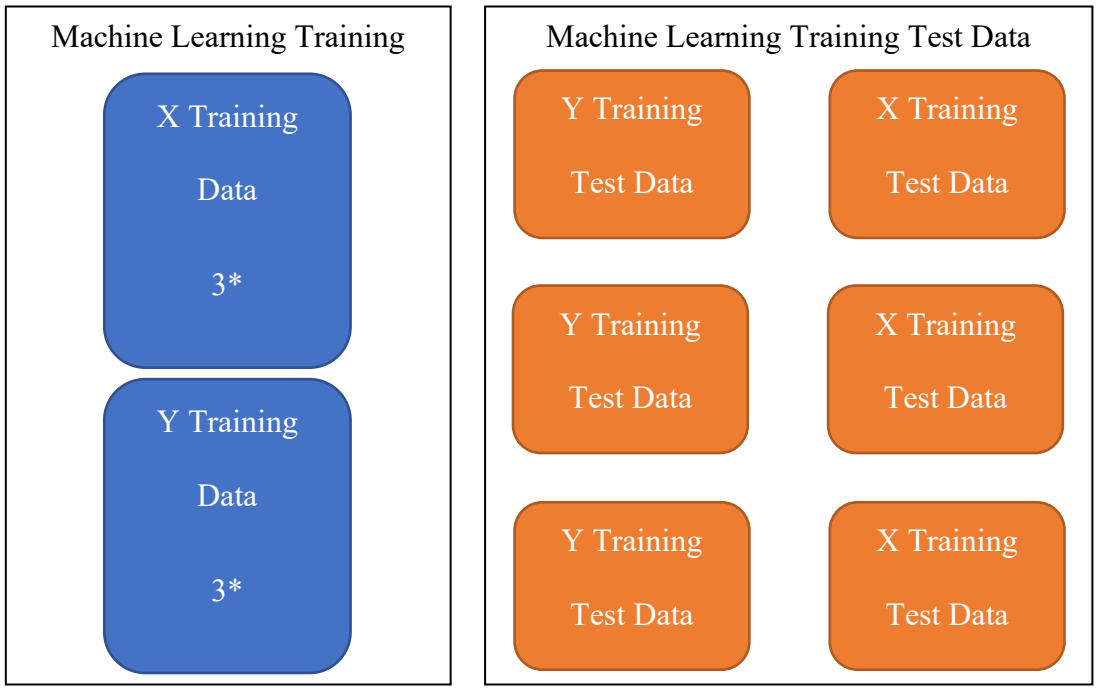


Figure 6-8 Hybrid mill training and test data structure

Once the axis matrices and the training test data matrices were made, the training of the RL and XAI models could begin.

6.3.3 Machine learning models for the prediction of dynamic errors of the hybrid mill

This section explains the machine learning model using the training data and how they operate within the digital twin to predict corrective actions for the hybrid mill.

Two machine learning algorithms are employed for this task: XAI and RL. These algorithms were selected to compare an interpretable model XAI with a traditional black-box approach, RL.

Regression Learning predicts outcomes based on historical data using a systematic approach. The specific mathematical formulation depends on the type of regression employed. Examples of the regression methods used include:

1. Linear Regression
2. Logistic Regression
3. Polynomial Regression
4. Ridge Regression
5. Lasso Regression
6. Quantile Regression
7. Bayesian Linear Regression
8. Principal Components Regression
9. Partial Least Squares Regression
10. Elastic Net Regression

A range of these regression learning options will be tested to determine the best response for predicting the dynamic error of the hybrid mill.

6.3.3.1 Regression Learning

When using regression learning, the choice of training method directly affects the accuracy and generalisability of the model. To identify the most effective approach, three base regression models were selected, and each was tested with seven variations, resulting in a total of multiple candidate models. These variations include different kernel functions, regularisation techniques, and optimisation settings to capture the dynamic behaviour of the hybrid mill. The regression learning process was conducted using MATLAB's Regression Learner app, where

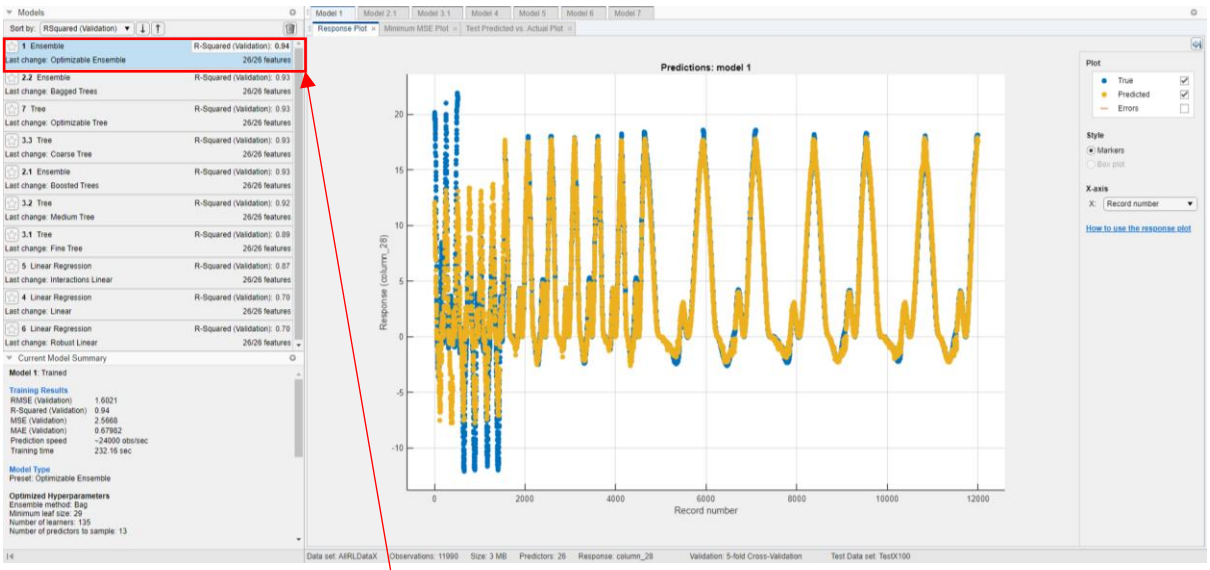
the input training matrix (including encoder data, acceleration key features, and feed rate) and the corresponding output data (ball bar measured errors) were loaded. Each model was trained, validated, and tested to determine its predictive performance. The best-performing regression model will then be used in the digital twin for real-time dynamic error prediction and corrective action.

The several types of regression learning used for training are:

- Tree
    - Coarse tree
    - Medium tree
    - Fine tree
    - Optimizable tree
- Linear
    - Linear
    - Interaction linear
    - Robust linear
- Ensemble
    - Bagged trees
    - Boosted trees
    - Optimizable ensemble

6.3.3.2 Training of the RL prediction model using X axis data

The X-axis training data was loaded into the regression learning app, and the training was conducted for the seven types used. The results of the training can be seen in Figure 6-9.



1 Ensemble	R-Squared (Validation): 0.94
Last change: Optimizable Ensemble	26/26 features
2.2 Ensemble	R-Squared (Validation): 0.93
Last change: Bagged Trees	26/26 features
7 Tree	R-Squared (Validation): 0.93
Last change: Optimizable Tree	26/26 features
3.3 Tree	R-Squared (Validation): 0.93
Last change: Coarse Tree	26/26 features
2.1 Ensemble	R-Squared (Validation): 0.93
Last change: Boosted Trees	26/26 features
3.2 Tree	R-Squared (Validation): 0.92
Last change: Medium Tree	26/26 features
3.1 Tree	R-Squared (Validation): 0.89
Last change: Fine Tree	26/26 features
5 Linear Regression	R-Squared (Validation): 0.87
Last change: Interactions Linear	26/26 features
4 Linear Regression	R-Squared (Validation): 0.70
Last change: Linear	26/26 features
6 Linear Regression	R-Squared (Validation): 0.70
Last change: Robust Linear	26/26 features

Figure 6-9 Hybrid mill regression learning training of X axis data

From the training results, it is clear that the Optimizable Ensemble model provided the best prediction performance across all training sets. During validation testing using the separate test

data, the model achieved a prediction accuracy of 94%. With such a high level of accuracy across the validation tests, the model for the X-axis error is expected to perform reliably when applied for real-time dynamic error prediction and corrective action.

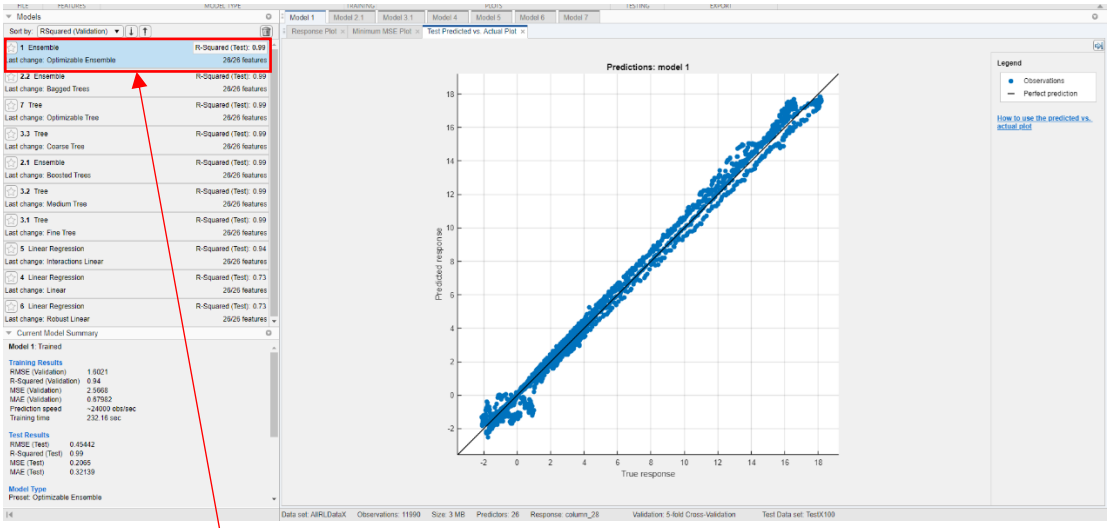
6.3.3.3 Evaluation of the RL dynamic error prediction model with new X-axis data

As previously mentioned, a fourth set of data was collected simultaneously with the training data to serve as an independent testing set. This testing data consists of one ball bar test at each feed rate: 500, 250, and 100 mm/min. The trained models will now be evaluated for their effectiveness using this unseen data.

To perform the testing, the data can be loaded into MATLAB's Regression Learning app, where the trained models will attempt to predict the ball bar errors. This allows the performance of the models to be assessed on new data. The testing data must be structured in the same format as the training data to ensure compatibility with the trained models.

6.3.3.3.1

Testing results of X axis RL dynamic error prediction model under 100 mm/min are shown in Figure 6-10. 99% prediction accuracy is observed,



1 Ensemble	R-Squared (Test): **0.99**
Last change: Optimizable Ensemble	26/26 features
2.2 Ensemble	R-Squared (Test): 0.99
Last change: Bagged Trees	26/26 features
7 Tree	R-Squared (Test): 0.99
Last change: Optimizable Tree	26/26 features
3.3 Tree	R-Squared (Test): 0.99
Last change: Coarse Tree	26/26 features
2.1 Ensemble	R-Squared (Test): 0.99
Last change: Boosted Trees	26/26 features
3.2 Tree	R-Squared (Test): 0.99
Last change: Medium Tree	26/26 features
3.1 Tree	R-Squared (Test): 0.99
Last change: Fine Tree	26/26 features
5 Linear Regression	R-Squared (Test): 0.94
Last change: Interactions Linear	26/26 features
4 Linear Regression	R-Squared (Test): 0.73
Last change: Linear	26/26 features
6 Linear Regression	R-Squared (Test): 0.73
Last change: Robust Linear	26/26 features

Figure 6-10 Evaluation result of the RL model under X-axis 100mm/min feed rate

Testing result of X-axis RL dynamic error prediction model under 250 mm/min (see Figure 6-18) shows a prediction accuracy of 97%.

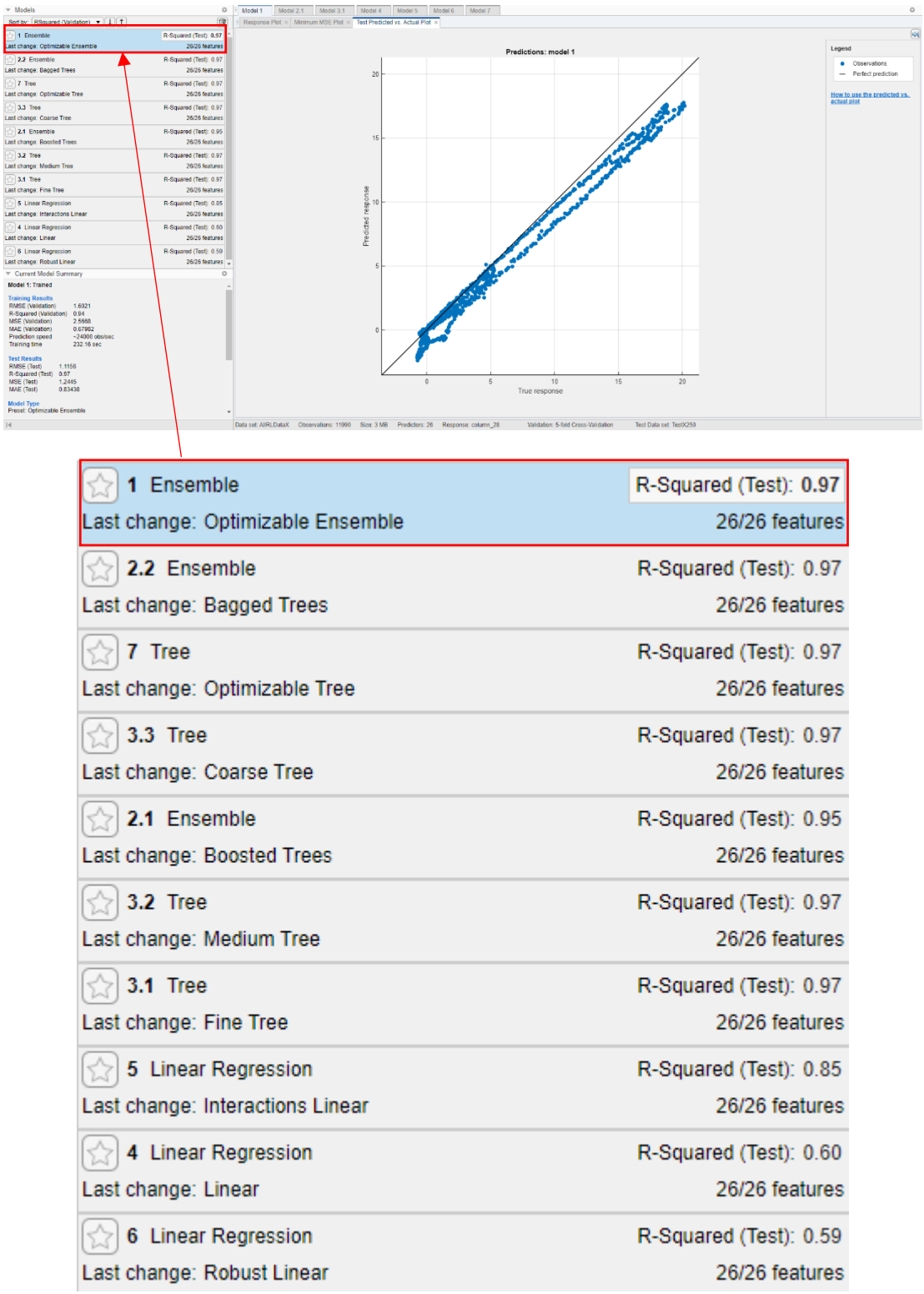
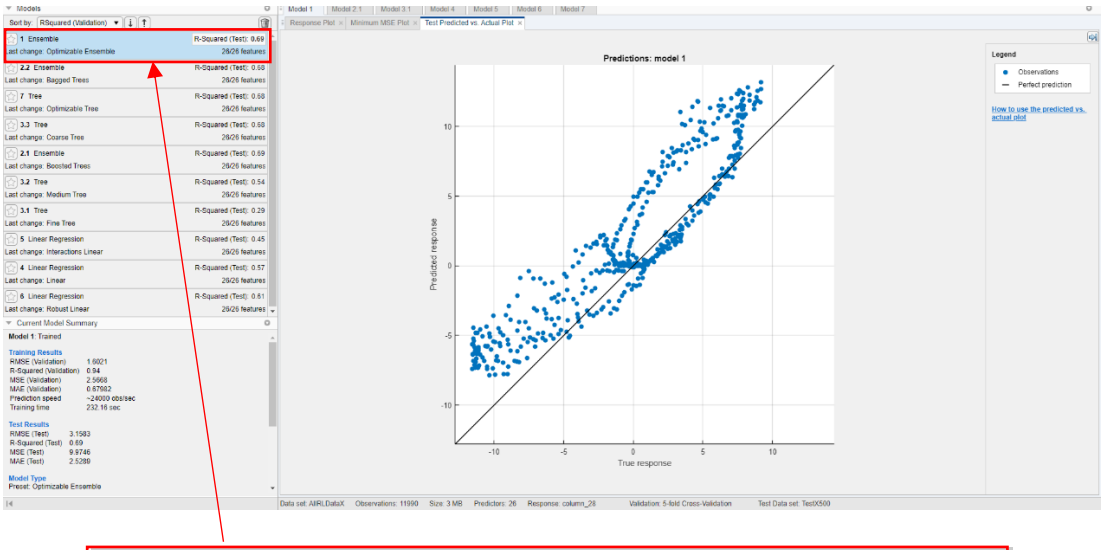


Figure 6-11 Evaluation result of the RL model under X-axis 2500mm/min feed rate

When the feed rate increases to 500 mm/min, the prediction accuracy drops to 69%, as shown in Figure 6-12.



1 Ensemble	R-Squared (Test): **0.69**
Last change: Optimizable Ensemble	26/26 features
2.2 Ensemble	R-Squared (Test): 0.68
Last change: Bagged Trees	26/26 features
7 Tree	R-Squared (Test): 0.68
Last change: Optimizable Tree	26/26 features
3.3 Tree	R-Squared (Test): 0.68
Last change: Coarse Tree	26/26 features
2.1 Ensemble	R-Squared (Test): 0.69
Last change: Boosted Trees	26/26 features
3.2 Tree	R-Squared (Test): 0.54
Last change: Medium Tree	26/26 features
3.1 Tree	R-Squared (Test): 0.29
Last change: Fine Tree	26/26 features
5 Linear Regression	R-Squared (Test): 0.45
Last change: Interactions Linear	26/26 features
4 Linear Regression	R-Squared (Test): 0.57
Last change: Linear	26/26 features
6 Linear Regression	R-Squared (Test): 0.61
Last change: Robust Linear	26/26 features

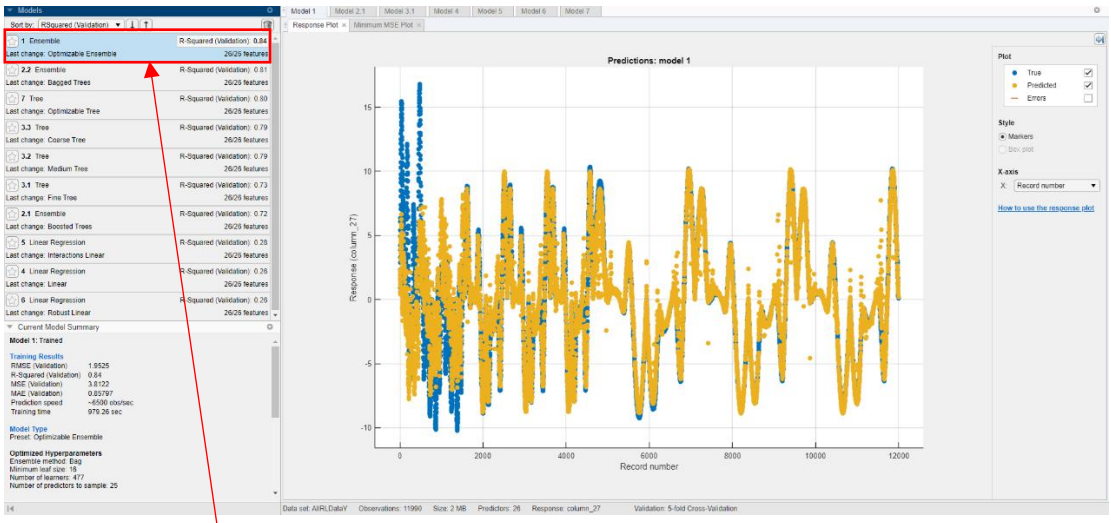
Figure 6-12 Evaluation result of the RL model under the X-axis 500mm/min feed rate

From the results of the testing data, it is evident that the regression learning model predicts ball bar errors with very high accuracy at lower feed rates of 100 and 250 mm/min, achieving 99% and 97% accuracy, respectively. This indicates that the model can reliably predict the dynamic

errors of the hybrid mill during operation without requiring a ball bar. However, the prediction accuracy decreases significantly at the higher feed rate of 500 mm/min, dropping to 69%. At this speed, the regression learning model's predictions may be less reliable and could potentially increase the magnitude of tracking errors.

6.3.3.4 Evaluation of the RL dynamic error prediction model with new Y-axis data

The Y-axis training data was loaded into the regression learning app, and the training was conducted for the same seven types as for the X data set. The result of the training is shown in Figure 6-13.

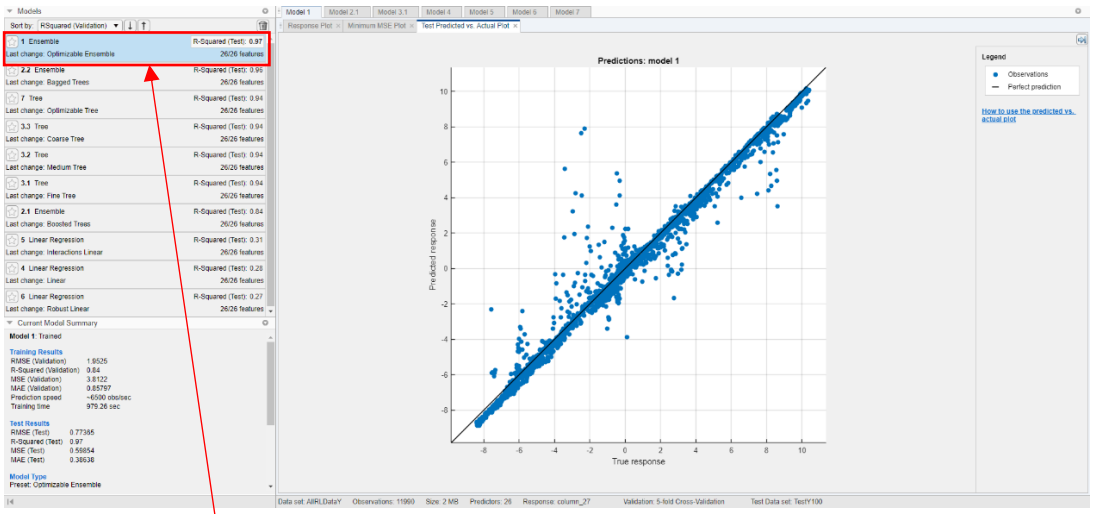


1 Ensemble	R-Squared (Validation): 0.84
Last change: Optimizable Ensemble	26/26 features
2.2 Ensemble	R-Squared (Validation): 0.81
Last change: Bagged Trees	26/26 features
7 Tree	R-Squared (Validation): 0.80
Last change: Optimizable Tree	26/26 features
3.3 Tree	R-Squared (Validation): 0.79
Last change: Coarse Tree	26/26 features
3.2 Tree	R-Squared (Validation): 0.79
Last change: Medium Tree	26/26 features
3.1 Tree	R-Squared (Validation): 0.73
Last change: Fine Tree	26/26 features
2.1 Ensemble	R-Squared (Validation): 0.72
Last change: Boosted Trees	26/26 features
5 Linear Regression	R-Squared (Validation): 0.28
Last change: Interactions Linear	26/26 features
4 Linear Regression	R-Squared (Validation): 0.26
Last change: Linear	26/26 features
6 Linear Regression	R-Squared (Validation): 0.26
Last change: Robust Linear	26/26 features

Figure 6-13 Training of the Y-axis RL dynamic error prediction model

The same training model for X-axis dynamic error prediction, the Optimizable Ensemble, was found to perform well for Y-axis dynamic error prediction, achieving an accuracy of 84%. While these results are somewhat lower than desired, they are largely affected by the larger

errors observed in the 500 mm/min tests. To better assess the model's effectiveness, it will next be evaluated using the Y-axis training test data, which will help determine whether the observed accuracy truly reflects the model's predictive capability.

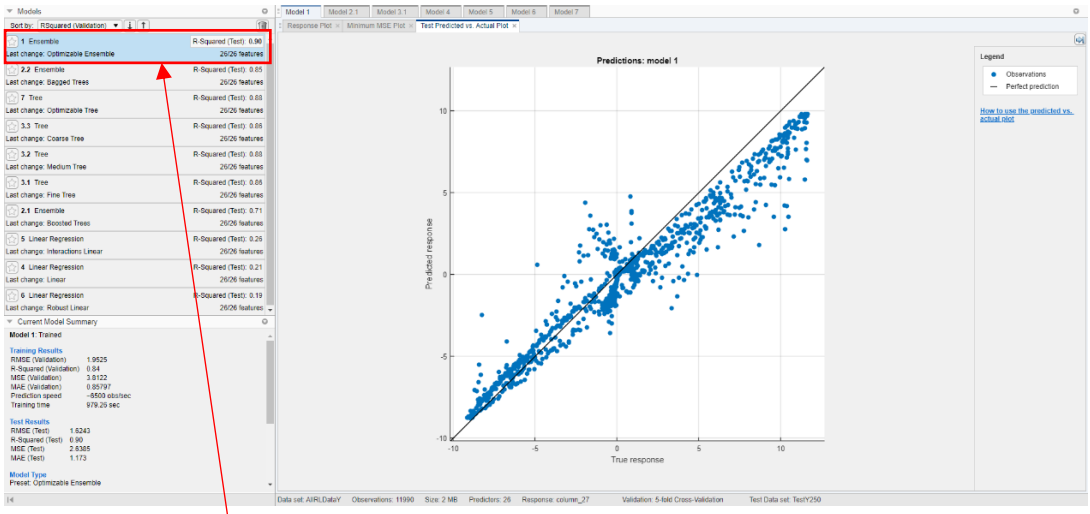


1 Ensemble	R-Squared (Test): **0.97**
Last change: Optimizable Ensemble	26/26 features
2.2 Ensemble	R-Squared (Test): 0.96
Last change: Bagged Trees	26/26 features
7 Tree	R-Squared (Test): 0.94
Last change: Optimizable Tree	26/26 features
3.3 Tree	R-Squared (Test): 0.94
Last change: Coarse Tree	26/26 features
3.2 Tree	R-Squared (Test): 0.94
Last change: Medium Tree	26/26 features
3.1 Tree	R-Squared (Test): 0.94
Last change: Fine Tree	26/26 features
2.1 Ensemble	R-Squared (Test): 0.84
Last change: Boosted Trees	26/26 features
5 Linear Regression	R-Squared (Test): 0.31
Last change: Interactions Linear	26/26 features
4 Linear Regression	R-Squared (Test): 0.28
Last change: Linear	26/26 features
6 Linear Regression	R-Squared (Test): 0.27
Last change: Robust Linear	26/26 features

Figure 6-14 Testing results of the Y-axis RL dynamic error prediction model under a feed rate of 100 mm/min

The rest of the results of the Y-axis RL dynamic prediction model (see Figure 6-14) show a prediction accuracy of 97% accuracy at a feed rate of 100 mm/min.

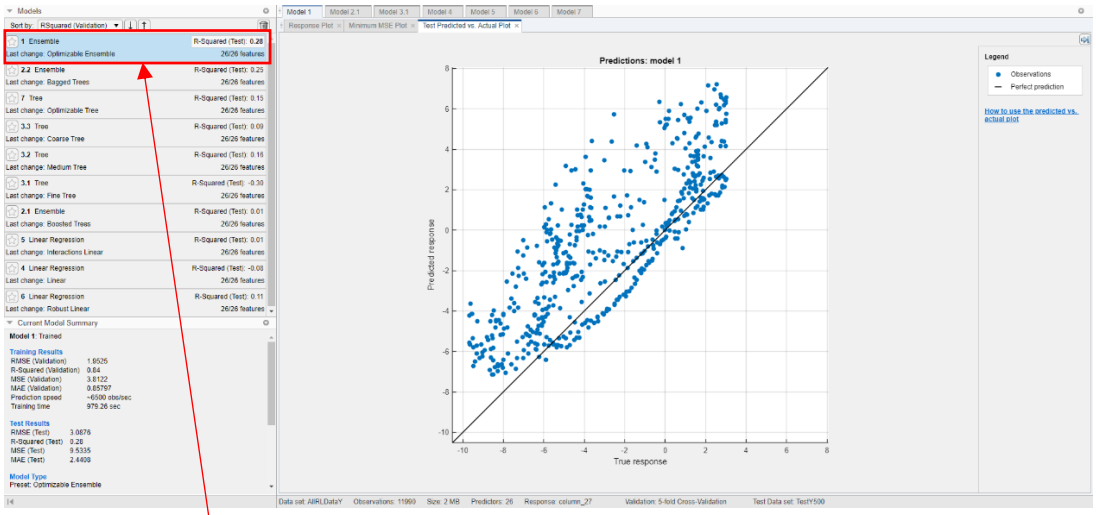
6.3.3.4.1 Y 250 mm/min model testing



1 Ensemble	R-Squared (Test): **0.90**
Last change: Optimizable Ensemble	26/26 features
2.2 Ensemble	R-Squared (Test): 0.85
Last change: Bagged Trees	26/26 features
7 Tree	R-Squared (Test): 0.88
Last change: Optimizable Tree	26/26 features
3.3 Tree	R-Squared (Test): 0.86
Last change: Coarse Tree	26/26 features
3.2 Tree	R-Squared (Test): 0.88
Last change: Medium Tree	26/26 features
3.1 Tree	R-Squared (Test): 0.86
Last change: Fine Tree	26/26 features
2.1 Ensemble	R-Squared (Test): 0.71
Last change: Boosted Trees	26/26 features
5 Linear Regression	R-Squared (Test): 0.26
Last change: Interactions Linear	26/26 features
4 Linear Regression	R-Squared (Test): 0.21
Last change: Linear	26/26 features
6 Linear Regression	R-Squared (Test): 0.19
Last change: Robust Linear	26/26 features

Figure 6-15 Testing results of the Y-axis RL dynamic error prediction model under a feed rate of 250 mm/min

A prediction accuracy of 90% is observed for the Y-axis RL model under a feed rate of 250 mm/min.



1 Ensemble	R-Squared (Test): **0.28**
Last change: Optimizable Ensemble	26/26 features
2.2 Ensemble	R-Squared (Test): 0.25
Last change: Bagged Trees	26/26 features
7 Tree	R-Squared (Test): 0.15
Last change: Optimizable Tree	26/26 features
3.3 Tree	R-Squared (Test): 0.09
Last change: Coarse Tree	26/26 features
3.2 Tree	R-Squared (Test): 0.16
Last change: Medium Tree	26/26 features
3.1 Tree	R-Squared (Test): -0.30
Last change: Fine Tree	26/26 features
2.1 Ensemble	R-Squared (Test): 0.01
Last change: Boosted Trees	26/26 features
5 Linear Regression	R-Squared (Test): 0.01
Last change: Interactions Linear	26/26 features
4 Linear Regression	R-Squared (Test): -0.08
Last change: Linear	26/26 features
6 Linear Regression	R-Squared (Test): 0.11
Last change: Robust Linear	26/26 features

Figure 6-16 Testing results of the Y-axis RL dynamic error prediction model under a feed rate of 500 mm/min

When the feed rate increases to 500100 mm/min, the prediction accuracy of the Y-axis RL prediction model is significantly reduced to 28%. This trend is similar to the result of the X-

axis RL prediction model. The cause of this reduction is not yet known and requires further investigation; insights from the XAI model may help explain this behaviour. For the slower feed rates, however, the results are promising, with prediction accuracies of 97% and 90% for 100 mm/min and 250 mm/min, respectively.

Overall, the performances of both the X-axis and Y-axis dynamic error RL prediction models at lower speeds indicate that accurate real-time dynamic error prediction is possible using the regression learning method. The average prediction accuracy across both axes at 100 mm/min and 250 mm/min is 95.75%, while the average accuracy across all tested feed rates is 80%. The lower overall accuracy is primarily due to the decreased performance at 500 mm/min, suggesting that higher feed rates may result in smaller improvements and could potentially induce errors. To improve predictions at higher speeds, additional training data across a wider range of feed rates or a faster sampling method may be required to better capture the dynamic signatures of the hybrid mill.

6.3.3.5 XAI prediction model

The XAI model was trained using the same dataset as the regression learning approach, processed in the same manner, and evaluated against the same fourth test set as the testing data. Unlike regression learning, which outputs predictions directly from the trained model, XAI generates an interpretable mathematical equation that predicts the dynamic error. This equation provides transparency into how the model derives its predictions. In this section, the trained XAI model will be presented, and its prediction performance will be reviewed and analysed.

6.3.3.5.1 XAI Training of X-axis data

Using the same training dataset as the regression learning approach, which included three feed rates with three repeated tests each, the QLattice XAI method generated a predictive equation to estimate the dynamic error measured by the ball bar. The prediction results are shown in

Figure 6-17 below, where the XAI model achieved a validation accuracy of 92% on the data it selected for validation.

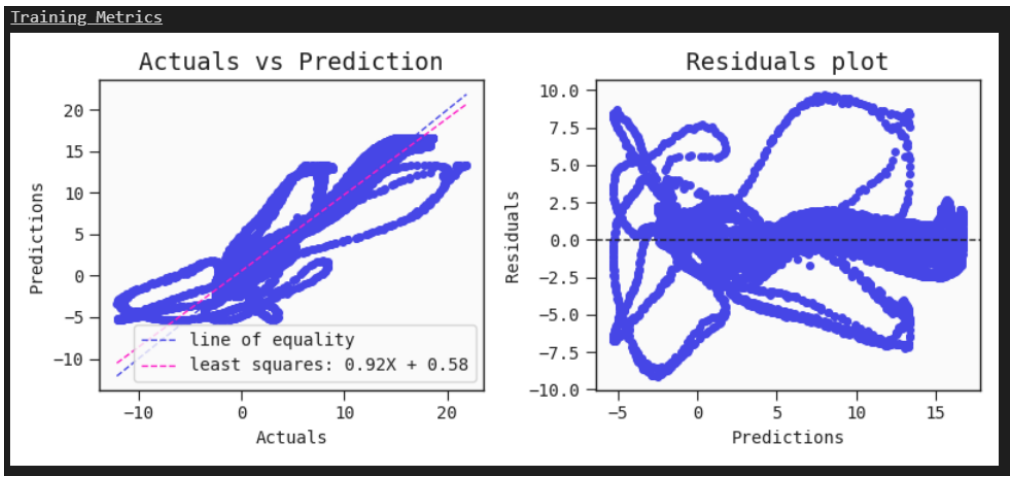


Figure 6-17 Hybrid mill XAI training of X-axis data

The prediction equation that was generated for this result is shown in Figure 6-18 and Equation 6-2, with its relationship between the key features of the matrix of data.

$$-0.181f_1 + 13.4e^{-0.467(-0.103f_5-1)^2-2.0\left(0.00113f_6+e^{-1.74(0.000786f_{16}-1)^2-0.0206(-0.0861f_2-1)^4}-0.111\right)^2} - 0.775$$

Figure 6-18 Hybrid mill XAI training of X-axis data equation

$$Sample = -0.181(F1) + 13.4e^{-0.467(-0.103(F5)-1)^2-2.0(0.00113(F6)+e^{-1.74(0.000786(F16)-1)^2-0.0206(-0.0861(F2)-1)^4}-0.111)^2} - 0.775 \quad (6\text{-}2)$$

Where F_1, F_2, F_5, F_6 and F_{16} are X Feedback, Y Feedback, X Position Error, Y Position Error, and RMS Accelerometer Table.

The generated equation 6-2 will be used in the testing of the X-axis on the fourth set of data to determine the reliability of the XAI-trained equation.

6.3.3.5.2 XAI Testing of X-axis model

Using the trained equation 6-2, the fourth data set will be tested to see how accurately the model can predict the ball bar results on new data that it has not seen. The results for the three feed rates are shown below.

100 X axis

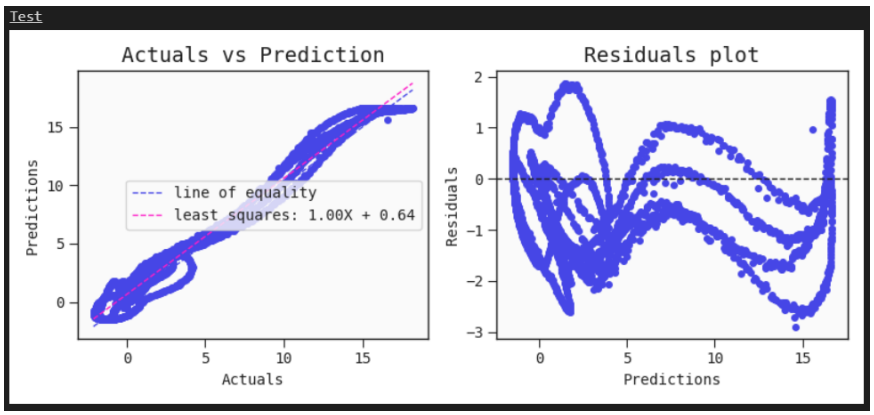


Figure 6-19 Hybrid mill XAI X 100 testing results

As shown in Figure 6-19 The prediction accuracy for the X axis at 100 mm/min shows a prediction accuracy of 100%.

250 X axis

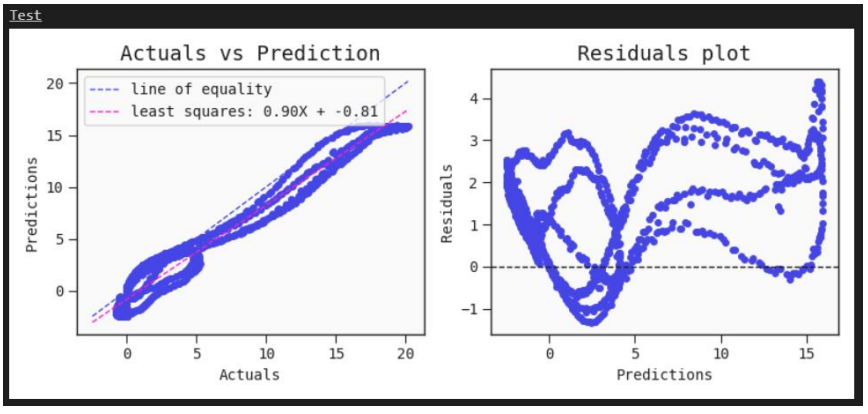


Figure 6-20 Hybrid mill XAI X 250 testing results

As shown in Figure 6-20 The prediction accuracy for the X axis at 250 mm/min shows a prediction accuracy of 90%.

500 X axis

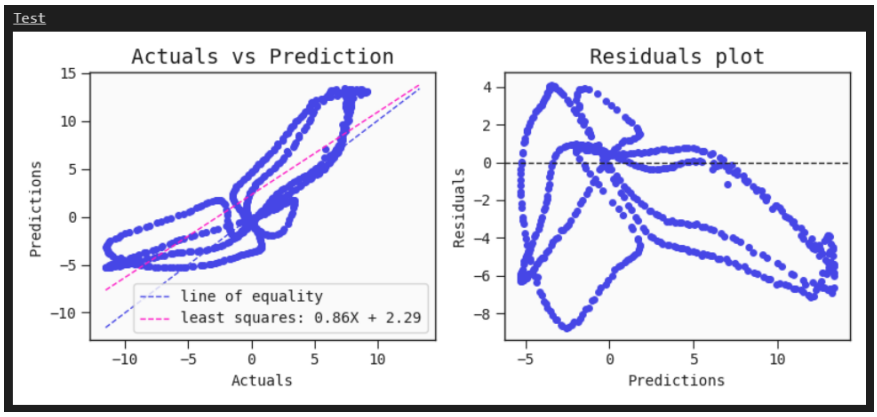


Figure 6-21 Hybrid mill XAI X 500 testing results

As shown in Figure 6-21 The prediction accuracy for the X axis at 500 mm/min shows a prediction accuracy of 86%.

From the fourth data set validation tests, the XAI can predict the ball bar error values across the range of feed rates with a high degree of accuracy. Then, the fourth data results are averaged the trained model can predict with a mean accuracy of 92%.

6.3.3.5.3 XAI Training of Y-axis data

The same Y-axis dataset used in the regression learning approach was applied to the XAI to generate an explainable machine learning model for predicting the ball bar dynamic error measurements of the Y axis of the hybrid mill. The training process followed the same procedure as for the X axis, and the model was tested on the Y-axis fourth dataset. The training results, shown in Figure 6-22, indicate a prediction accuracy of 48%, which is lower than expected. To verify this outcome, the data and training process were double-checked and

repeated, yielding similar results.

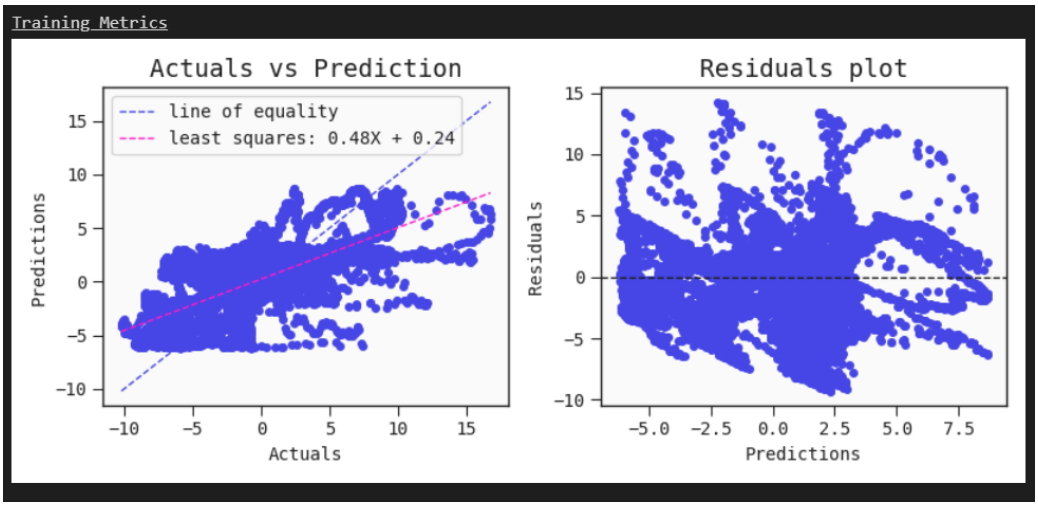


Figure 6-22 Hybrid mill XAI training of Y-axis data

For this XAI prediction model, the relationship between the key features and Equation 6-1 are shown in Figure 6-23.

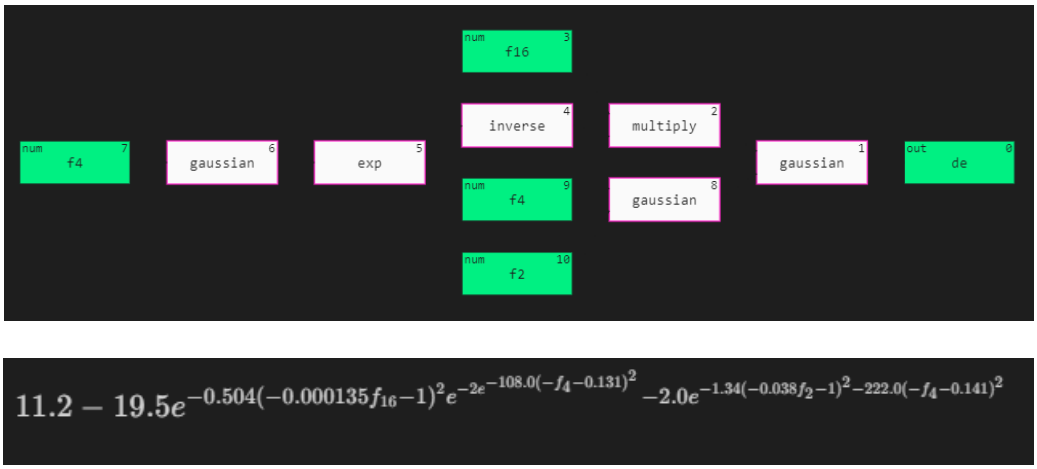


Figure 6-23 Hybrid mill XAI training of Y axis data equation

$$Y\ Error\ Prediction = 11.2 - 19.5e^{-0.504(-0.000135(F16)-1)^2 * e^{-2e^{-108.0(-(F4)-0.131)^2}} - 2.0e^{-1.34(-0.038(F2)-1)^2-222.0(-(F4)-0.141)^2}} \quad (6\text{-}3)$$

Equation 6-1 Hybrid Mill XAI Y-axis error prediction

Using this equation, the fourth data set will be tested on the model to see how effective it is with new data.

6.3.3.5.4 XAI Testing of Y-axis model

100 Y Axis

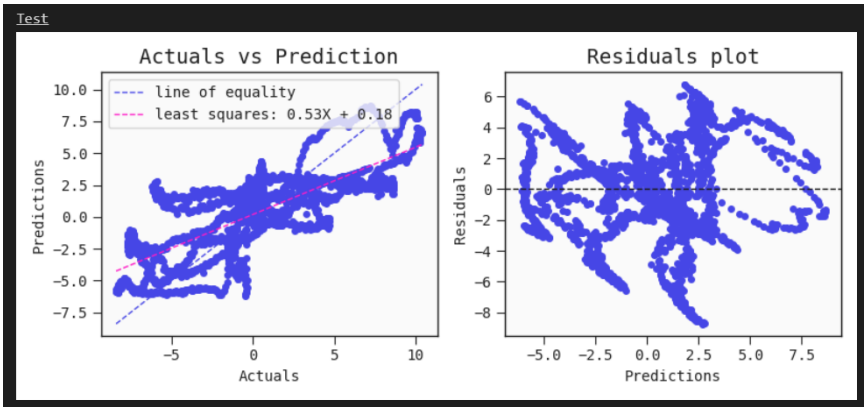


Figure 6-24 Hybrid mill XAI Y 100 testing results

As shown in Figure 6-24 The prediction accuracy for the Y axis at 100 mm/min shows a prediction accuracy of 53%.

250 Y axis

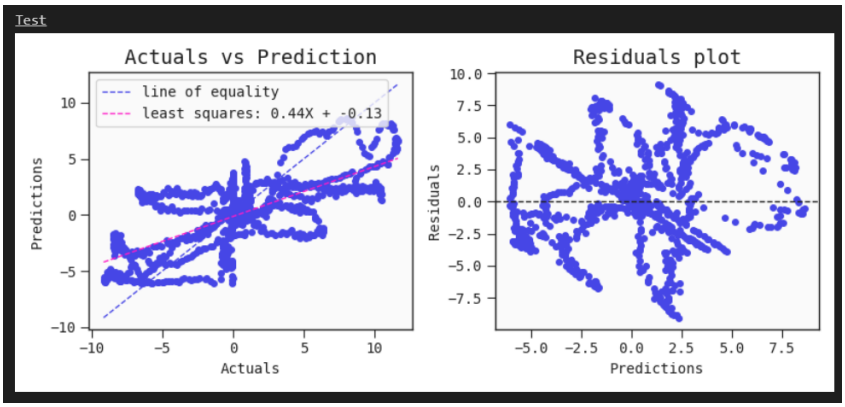


Figure 6-25 Hybrid mill XAI Y 250 testing results

As shown in Figure 6-25 The prediction accuracy for the Y axis at 250 mm/min shows a prediction accuracy of 44%.

500 Y axis

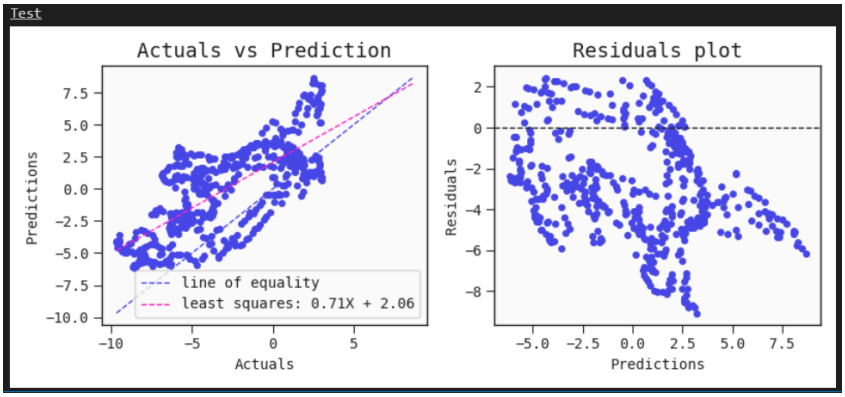


Figure 6-26 Hybrid mill XAI Y 500 testing results

As shown in Figure 6-26 The prediction accuracy for the Y axis at 500 mm/min shows a prediction accuracy of 71%.

From the validation tests using the fourth dataset, it is evident that the XAI model is also struggling to predict the ball bar results for the Y axis, showing low prediction accuracy similar to the regression learning model. The XAI's own validation accuracy for the Y axis was 48%, and for the 100 mm/min and 250 mm/min feed rates, the prediction accuracies were 54% and 44%, respectively. Interestingly, the accuracy for the 500 mm/min feed rate increased significantly to 71%, indicating that for this feed rate, the regression learning model may perform better. The cause of the overall reduction in accuracy is unclear, given that the same data was used for both training approaches. This may reflect a limitation of the XAI method for this specific task, with an average prediction accuracy for the Y axis of 56%.

6.3.3.6 Test accuracy results table

The results of the training and testing of the models are shown in Table 6-3 Below the training results, the fourth data testing, averages by axis and average by model are shown.

Table 6-3 Hybrid mill machine learning testing results

	Regression Learning (prediction Accuracy %)		**XAI (prediction Accuracy %)**	
	X	Y	X	Y
Training validation Results (%)	94	84	92	48
Fourth 100 data (mm/min)	99	97	100	53
Fourth 250 data (mm/min)	97	90	90	44
Fourth 500 data (mm/min)	69	28	86	71
Average prediction accuracy by axis on test data	88.3	71.6	92	56

Average prediction accuracy by model	80	74

From reviewing the training and testing results, it is clear that both models perform at a similar level of accuracy, with only a 6% difference in prediction between them. Another point of interest is the reduced prediction accuracy for the Y-axis dataset compared to the X-axis. This discrepancy is not fully understood, as the same data collection, processing, and training methods were applied to both axes. One possible explanation is the structural design of the hybrid mill, where the Y-axis sits on the X-axis. Information from both axes may be required to accurately predict Y-axis dynamic errors; this hypothesis will need further investigation.

Given the similar prediction accuracy of both models, they will be tested on the hybrid mill to evaluate their ability to accurately predict and correct dynamic errors in real-time operation.

6.3.4 Hybrid mill corrective action methodology

The error prediction models that have been developed allow for the application of predicted corrections in real time. This approach is necessary to achieve micrometre corrections during operation. The compensation method utilises encoder offsets, which can be dynamically adjusted by the A3200 controller in real time based on an analogue input voltage. The command used to implement this corrective action is shown in Figure 6-27, with a detailed breakdown of its components provided.

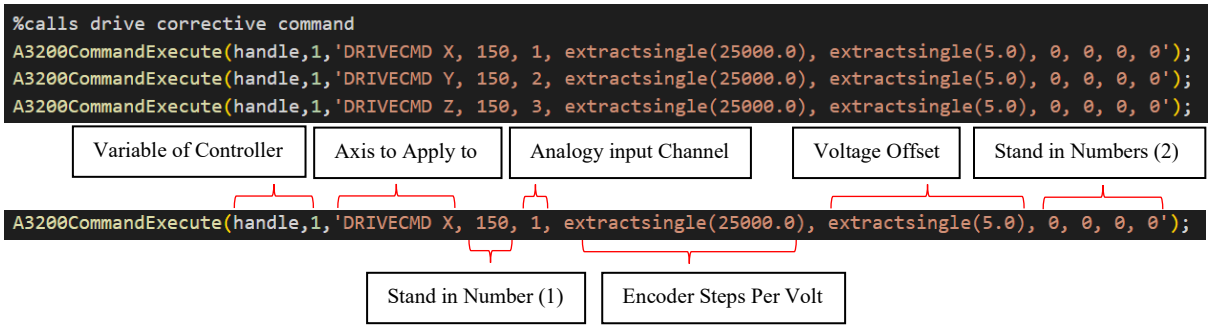


Figure 6-27 Aerotech A3200 corrective action command breakdown

The command requires a number of variables to be set up correctly for each axis. These parts are:

•**Variable of Controller** – Specifies the variable used to reference the A3200 controller when called in MATLAB.

•**Axis to Apply To** – Determines which axis will receive the corrective command.

•**Stand-in Number (1)** – Required by the Aerotech command syntax; acts as a placeholder for unused variables.

•**Analogue Input Channel** – Selects the analogue input channel that reads the signal to apply the corrective action.

•**Encoder Steps per Volt** – Sets the number of encoder steps corresponding to one volt applied to the analogue input. Currently configured as 25 µm per volt, allowing ±125 µm of correction.

•**Voltage Offset** – Applies an offset to the measured analogue voltage before computing the correction.

•**Stand-in Numbers (2)** – Additional placeholders required by the command syntax for unused variables

This command can be applied to any axis of the hybrid mill, enabling real-time positional offsets based on the measured analogue voltage. By applying an offset to the encoder value,

the controller adjusts the axis position without modifying the G-code, effectively allowing on-the-fly corrections. Furthermore, by setting the offset to the midpoint of the analogue input range, both positive and negative corrective actions can be applied: an increase in voltage results in a positive correction, while a decrease produces a negative correction. Figure 6-28 illustrates this operation in practice.

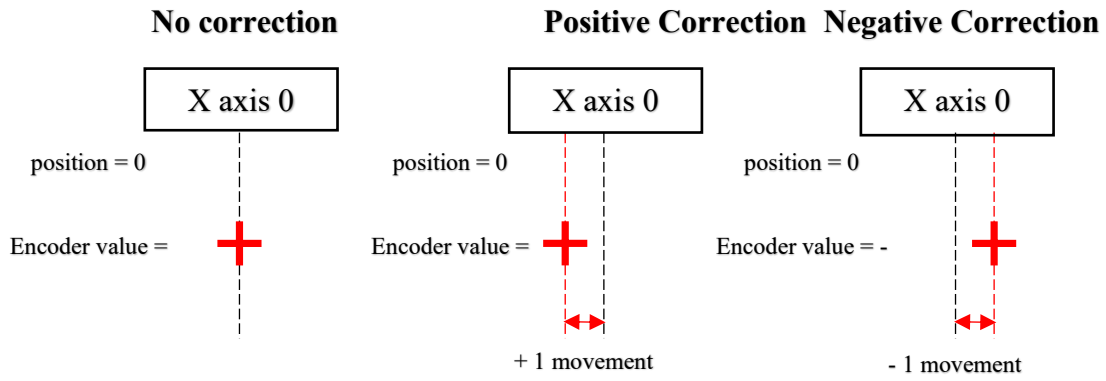


Figure 6-28 Aerotech corrective action explanation

To validate the effectiveness of the dynamic error compensation method, an experiment was conducted to verify the axis movement resulting from the Aerotech command. In this test, a known voltage was applied to the analogue input using a bench power supply, set to 0 V, 5 V, and 10 V. The Aerotech command was active, causing the X-axis to apply the corresponding encoder offset while a ball bar was set up to record live positional changes. This setup allowed the corrective action to be measured directly by the ball bar as a function of the applied voltage. For this validation, the steps-per-volt parameter was set to 1000, enabling finer resolution for smaller positional adjustments. The results of this validation are shown in Figure 6-29.

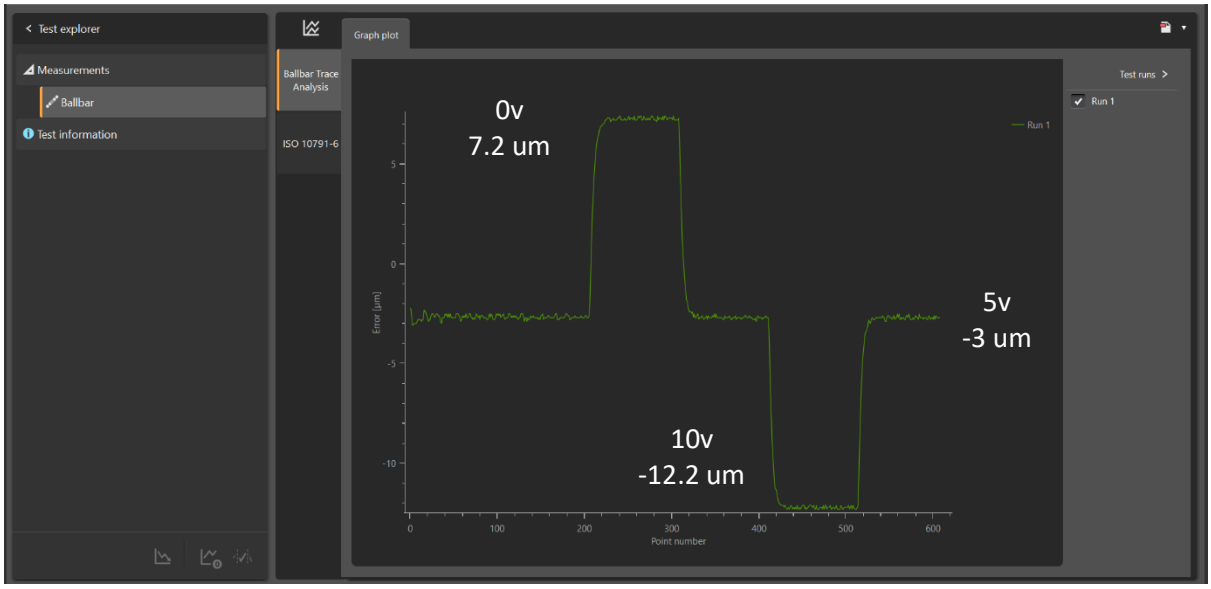


Figure 6-29 Aerotech corrective action validation test

From the results, it is evident that the mill's X-axis position changes in response to the Aerotech command and the voltage applied to the analogue input channel. The minor positioning errors observed are likely due to small inaccuracies in the bench power supply voltage, but overall, this experiment validates that the command functions correctly and that the mill's position can be adjusted in real time via the analogue input.

6.3.5 Implementation of a digital twin -driven dynamic error compensation approach on a hybrid mill

To implement the proposed approach on the hybrid milling machine, it requires generating and applying variable control voltage to the analogue channels of the Aerotech A3200. An Arduino was used for this purpose, communicating with MATLAB through the Arduino Support Package, which allows reading and writing of digital signals. Since the Aerotech A3200 analogue inputs accept 0–10 V, a PWM-to-voltage conversion board is used to convert the Arduino's PWM signals into fixed voltages. These boards also step up the voltage to cover the full 0–10 V range, as the Arduino itself is limited to 5 V output. One board is installed for each axis requiring corrective action, meaning three boards are used for the X, Y, and Z axes. The

wiring diagram for the three PWM-to-voltage boards and the connection with the correction voltage through the Arduino Uno is illustrated in Figure 6-30.

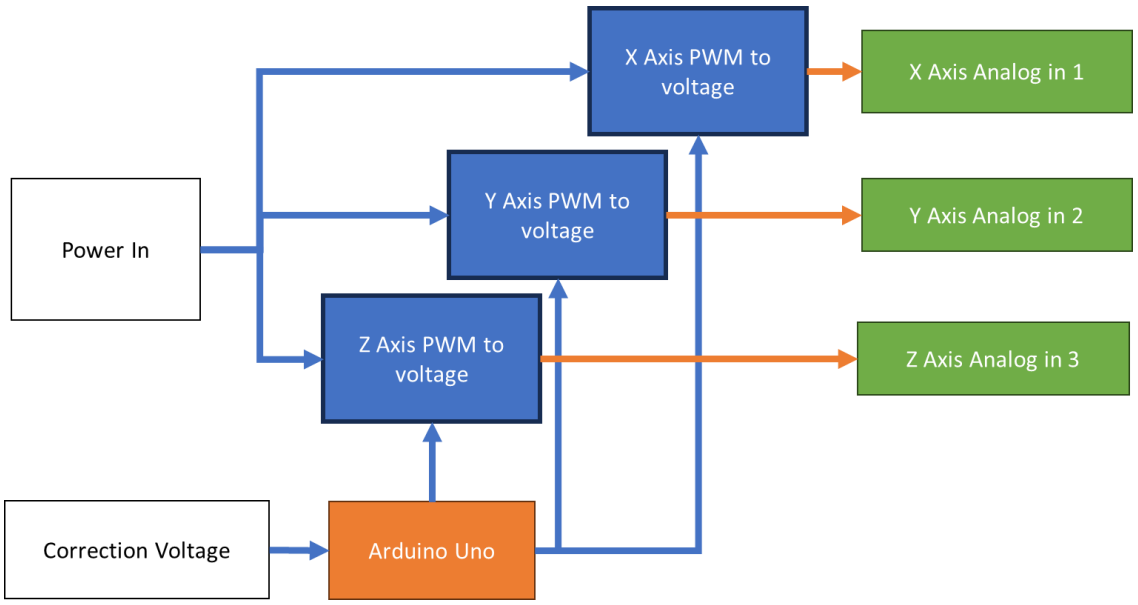


Figure 6-30 Hybrid mill PWM to voltage wiring diagram for corrective action

Each PWM-to-voltage board has four inputs and two outputs. The inputs consist of the PWM signal positive and negative terminals (+, −) and the power supply inputs (+, −). The outputs provide the converted voltage corresponding to the PWM input, which is then fed to the A3200 analogue input for real-time corrective action.

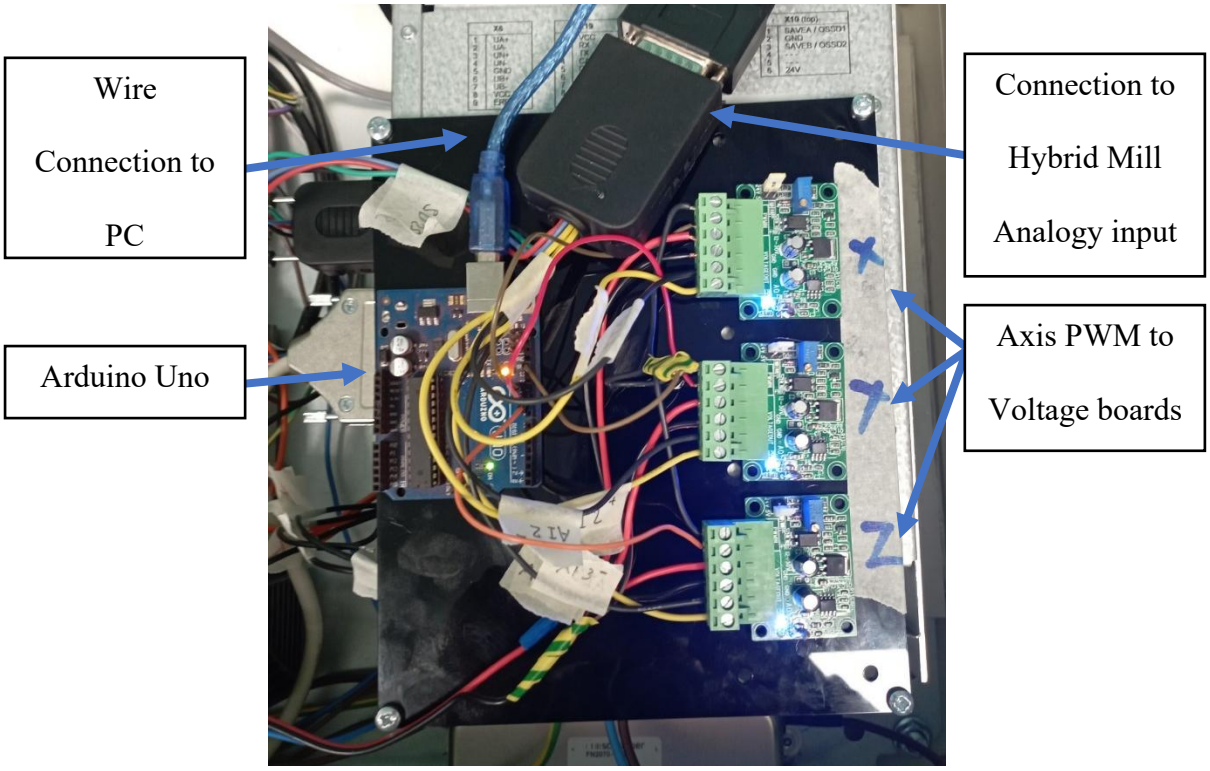


Figure 6-31 Connective PWM to voltage for the hybrid mill

The electrical components were mounted on a plastic board to facilitate installation and provide electrical isolation from the rest of the hybrid mill control box. The installation of this setup is shown in Figure 6-38. Once connected, the voltage output of each PWM-to-voltage board can be adjusted using the onboard potentiometer. Calibration is performed for each axis so that a PWM input of 127 from the Arduino corresponds to a 5 V output. This calibration allows the Arduino to generate a full 0–10 V range for the hybrid mill. The results of the calibration are shown in Figure 6-32, where the voltage outputs for channels 1, 2, and 3 (corresponding to the X, Y, and Z axes) are all set to 5 V with an accuracy of 4.5 mV.

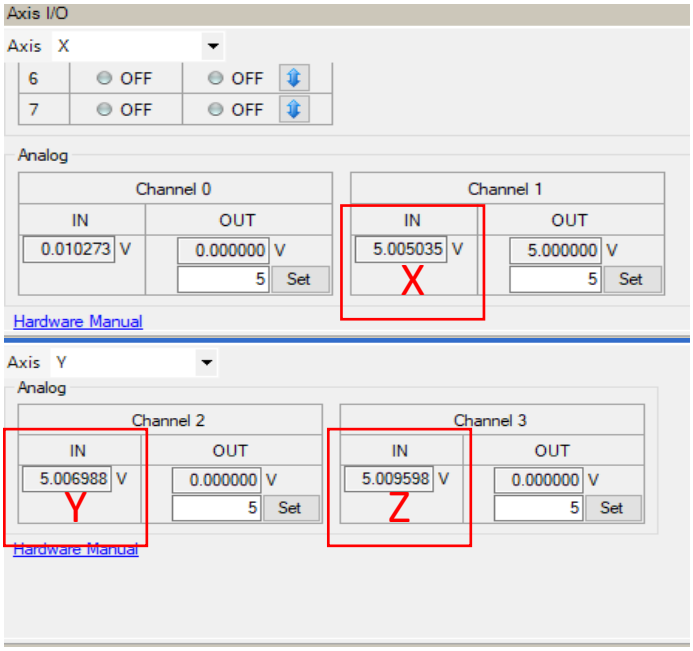


Figure 6-32 Hybrid mill calibration of input voltages

With all the electronics added to the hybrid mill case, the flow chart of the procedure to implement the corrective action is shown in Figure 6-330.

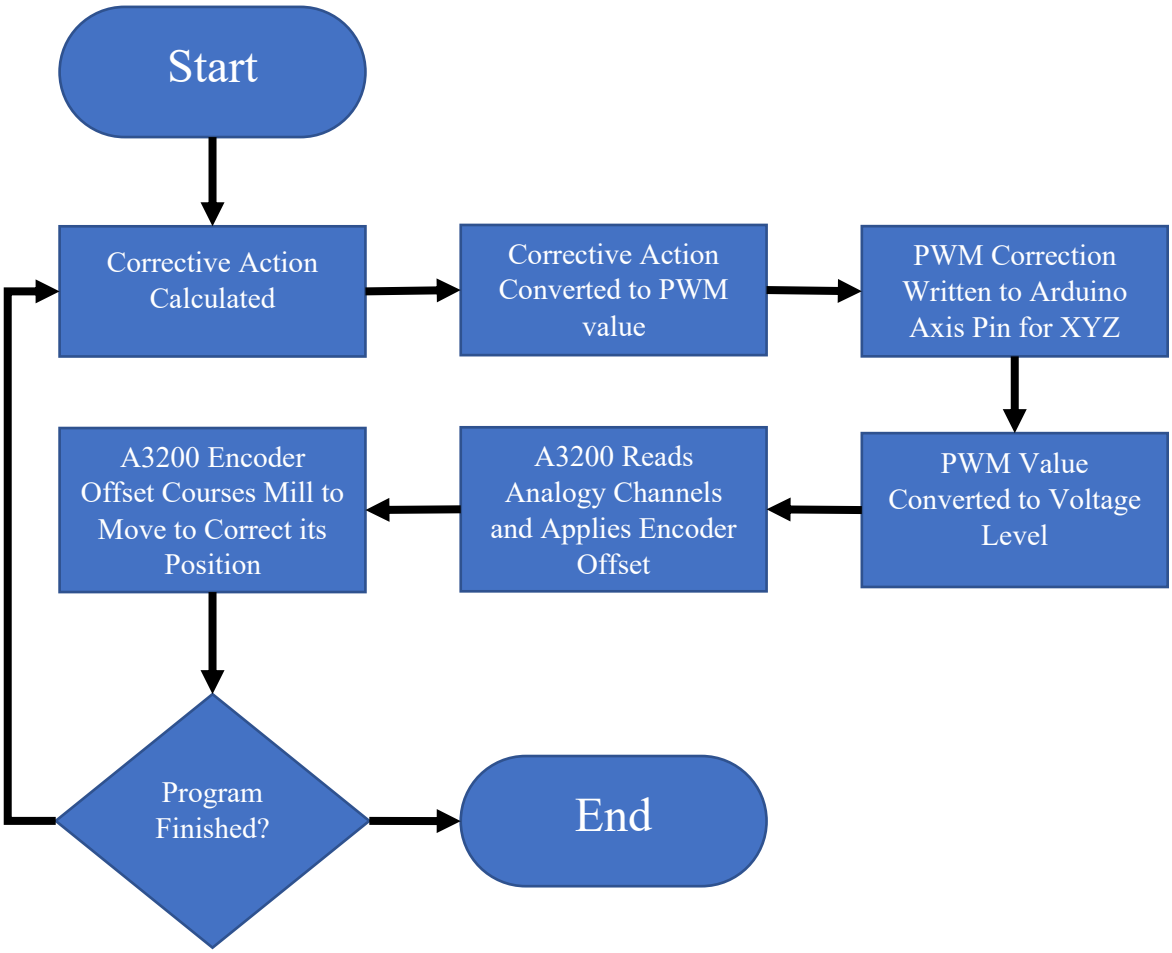


Figure 6-33 Hybrid mill corrective action process

6.3.6 Validation of the error correction hardware and programme

To further validate that the wiring and programming for the hybrid mill corrective action are functioning correctly, three ball bar tests were conducted with different voltages applied to the analogue inputs from the Arduino. These tests were performed in the XY plane at a feed rate of 1000 mm/min, minimising the influence of dynamic errors on the measurements and isolating the effect of the applied voltage on the mill's position. The result of the three tests can be seen in Figure 6-34, with the Ball Bar 20 software results shown in Figure 6-35.

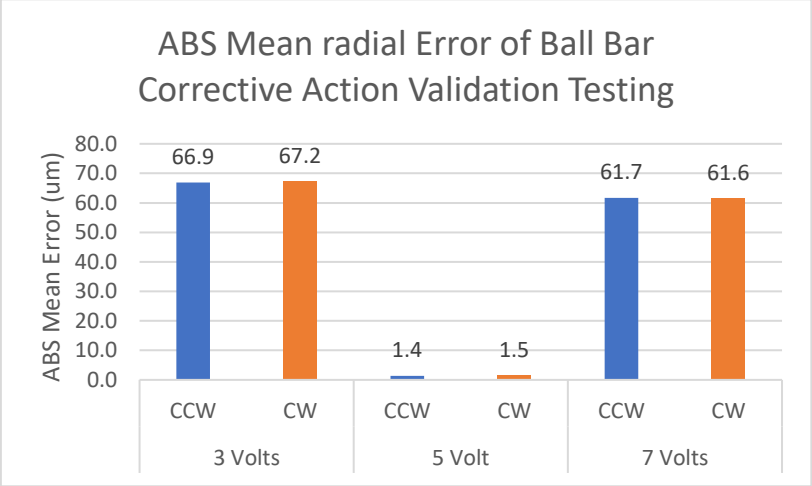


Figure 6-34 Hybrid mill ball bar corrective action validation

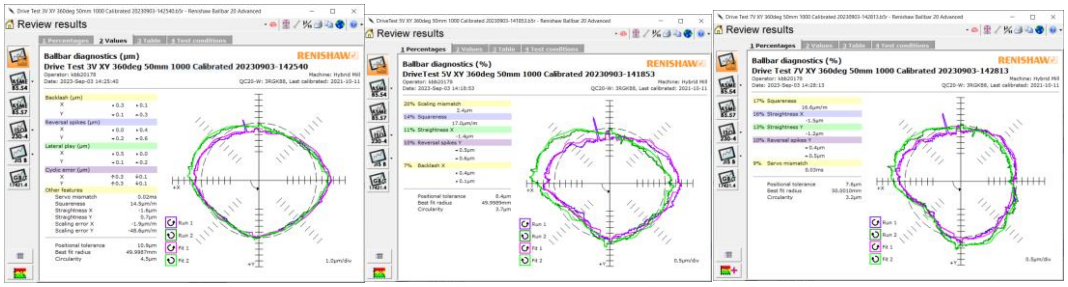


Figure 6-35 Hybrid mill ball bar 20 results

As shown in Figure 6-41, the absolute mean radial error measured by the ball bar differs from the no-correction baseline (5 V) by nearly equal amounts for the 3 V and 7 V tests, which is consistent with expectations. This indicates that the potentiometer could be further fine-tuned to achieve more consistent corrective movements in both positive and negative directions, or

the resolution of the correction could be improved by reducing the step size of each PWM increment. This would increase the precision of the 8-bit Arduino PWM signal, allowing for smaller and more accurate corrective adjustments.

With the corrective action method successfully demonstrated, the next step is to conduct the experiment to evaluate the effectiveness of machine learning–based dynamic error compensation action on the hybrid mill, to determine whether this approach can improve its dynamic working performance.

6.4 Experimental procedure for evaluating the proposed dynamic error compensation approach

The experiment to evaluate the effectiveness of the digital twin for dynamic error prediction on the hybrid mill will be conducted using three ball bar tests at feed rates of 100, 250, and 500 mm/min, respectively, both with and without corrective action, employing the best-performing RL and XAI models. Each test will be repeated three times, resulting in a total of 36 ball bar tests to compare the two dynamic error prediction methods. The findings from these tests will determine which method provides the most accurate prediction of the hybrid mill's dynamic errors and will inform the selection of the model to be used in developing a combined digital twin for smart manufacturing applications. The full set of planned tests is summarised in Table 6-4.

Table 6-4 Hybrid mill XAI and RL validation test conditions

Regression Learning						
500 mm/min	Test 1 WC	Test 2 WC	Test 3 WC	Test 1 NC	Test 2 NC	Test 3 NC
250 mm/min	Test 1 WC	Test 2 WC	Test 3 WC	Test 1 NC	Test 2 NC	Test 3 NC
100 mm/min	Test 1 WC	Test 2 WC	Test 3 WC	Test 1 NC	Test 2 NC	Test 3 NC
Explainable AI						
500 mm/min	Test 1 WC	Test 2 WC	Test 3 WC	Test 1 NC	Test 2 NC	Test 3 NC
250 mm/min	Test 1 WC	Test 2 WC	Test 3 WC	Test 1 NC	Test 2 NC	Test 3 NC
100 mm/min	Test 1 WC	Test 2 WC	Test 3 WC	Test 1 NC	Test 2 NC	Test 3 NC

The setup and positioning of the ball bar test will replicate the conditions used for the training data. This ensures that the live readings during testing are as similar as possible to the training data, allowing the machine learning models to generate accurate predictions and enabling identification of the most reliable model. The setup, including the ball bar and the accelerometers mounted on the table and spindle, is the same as previous hybrid mill ball bar tests.

The third accelerometer, used to measure background vibrations, is mounted to the frame of the hybrid mill and is not visible in the photo. The same starting point will be used for all tests, ensuring that any static errors remain consistent across trials. To minimise the effects of temperature on the hybrid mill, all tests will be conducted consecutively. If the mill has not been in operation before testing, a warm-up routine will be performed. This routine consists of the same ball bar motion, run at a feed rate of 3000 mm/min and repeated three times, to warm up the bearings and actuators and reduce the influence of thermal expansion on performance.

6.5 Results of evaluation of the predictive digital twin-driven dynamic error compensation approach for the hybrid mill

The absolute mean error for the X and Y axes is used to assess the accuracy of the proposed dynamic error compensation outcomes. The errors from both axes will be summed to provide the total positioning error of the hybrid mill. This method accounts for both positive and negative deviations during the ball bar test and allows for a clear comparison of the performance of each axis at each feed rate. The results for all tests are presented in Table 6-5, with the average errors for each feed rate summarised in Table 6-6.

Table 6-5 Evaluation test results for XAL and RL-based dynamic error compensations

Test Number	**Test 1**				**Test 2**				**Test 3**			
Corrective action	**With Compensation**		**With Compensation**		**With Compensation**		**Without Compensation**		**Without Compensation**		**Without Compensation**	
Axis Tested	X	Y	X	Y	X	Y	X	Y	X	Y	X	Y
Regression Learning												
500 mm/min	4.88	3.86	5.18	4.00	5.09	3.97	5.31	5.59	5.26	5.58	5.20	5.54
250 mm/min	9.29	6.82	9.24	6.78	9.24	6.79	5.47	5.79	5.41	5.78	5.42	5.79
100 mm/min	26.69	9.63	26.62	9.95	26.45	10.29	3.63	2.74	3.53	2.61	3.49	2.56
Explainable AI												
500 mm/min	6.58	9.90	6.44	9.90	6.36	9.92	5.31	5.59	5.26	5.58	5.20	5.54
250 mm/min	8.18	12.90	10.36	14.87	10.38	14.86	5.47	5.79	5.41	5.78	5.42	5.79
100 mm/min	15.69	39.39	15.66	39.47	15.66	39.46	3.63	2.74	3.53	2.61	3.49	2.56

Table 6-6 Hybrid mill XAI and RL validation test results averaged

	Average RL		Average XAI		Error Reduction	
	WC	NC	WC	NC	RL	XAI
500 mm/min	9.61	9.74	14.58	14.14	**1.30**	**-3.11**
250 mm/min	14.47	13.18	19.19	18.27	**-9.83**	**-5.05**
100 mm/min	26.42	18.70	38.86	34.81	**-41.28**	**-11.64**

As shown in Table 6-6, the proposed error compensation approaches increase the hybrid mill's positioning error, with the largest increase reaching −41% at the feed rate of 100 mm/min. The only improvement was observed at 500 mm/min, where the RL model achieved a modest error reduction of 1.3%. A clearer interpretation of these results is provided in the graphical representations: the RL model results are shown in Figure 6-36 with a further breakdown by machine axis in Figure 6-37, and the XAI results are presented in Figure 6-38 with a further breakdown by axis in Figure 6-39.

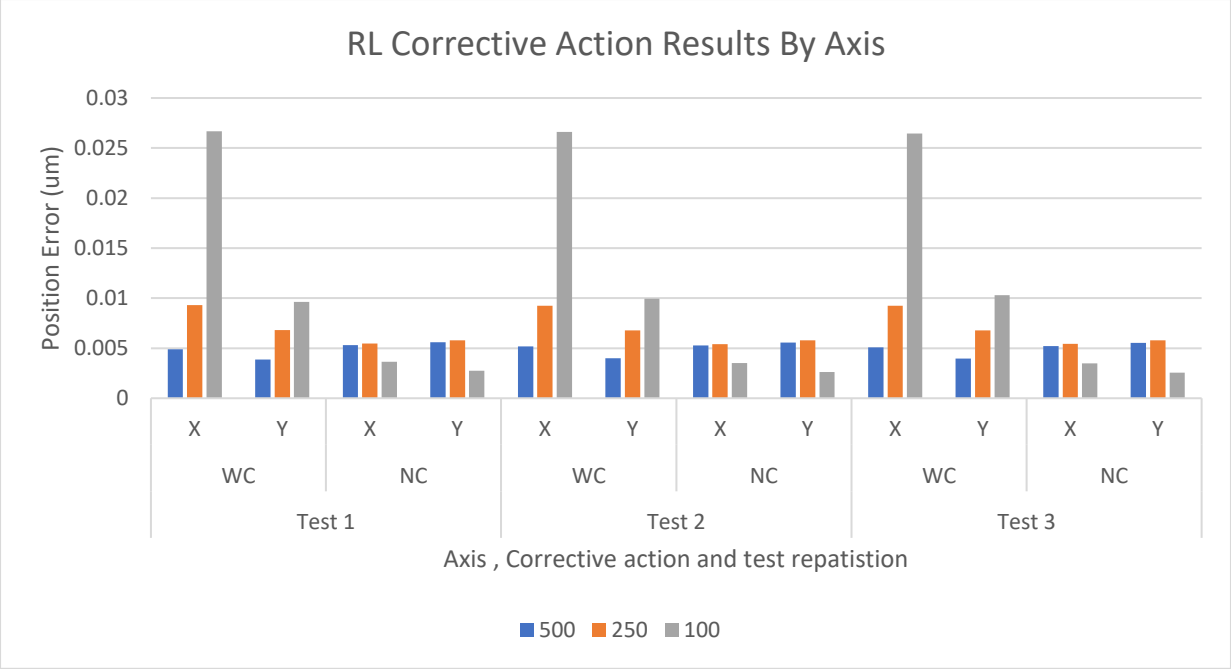


Figure 6-36 RL corrective action results by axis

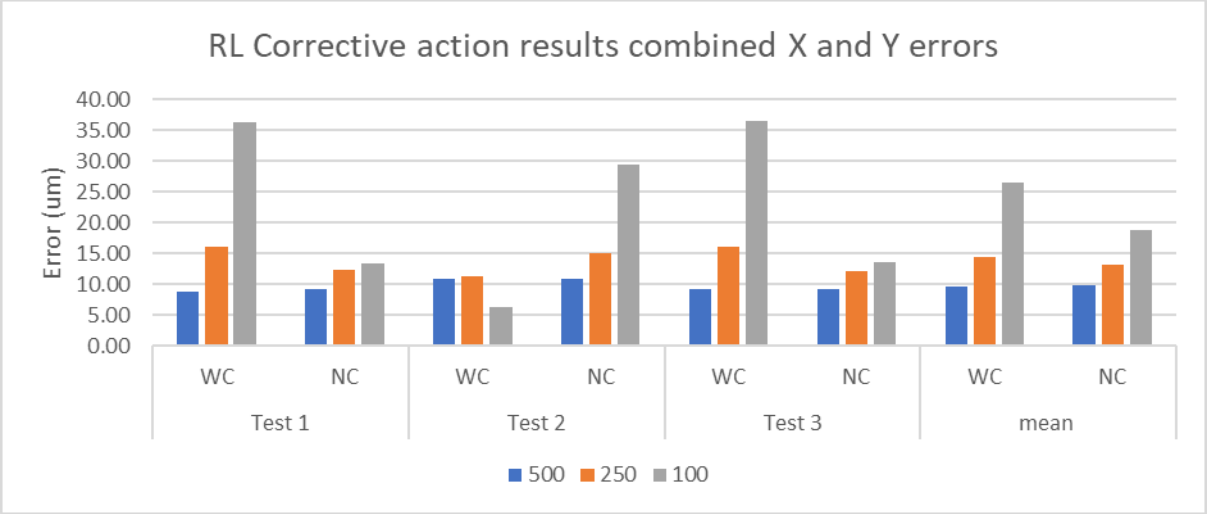


Figure 6-37 RL corrective action results combined

The results from the RL model show an unexpected trend: The X-axis compensation tests produced greater errors than the Y-axis, despite the X-axis having a higher prediction accuracy during validation (88.3% vs. 71.6% for the Y-axis). This suggests that although the model predictions were accurate in theory, their application as corrective actions may not have been fully effective. A detailed analysis of the predicted correction magnitudes during the test could

help identify whether the issue arises from overshooting, timing delays, or limitations in the actuator resolution.

Interestingly, the 500 mm/min tests consistently showed error reduction across all repetitions, indicating that the model performs better at higher feed rates. One possible explanation is that larger dynamic errors at higher speeds are easier for the accelerometer system to detect and for the correction algorithm to compensate, whereas at lower speeds, the error may be close to the resolution limit of the acceleration and actuation system.

The inconsistent performance at lower feed rates is particularly noteworthy. Validation tests suggested high prediction accuracy (>90%) at 100 and 250 mm/min, yet these accuracies did not translate into effective real-time error reduction. This discrepancy may indicate that the model is sensitive to subtle dynamic effects, sensor noise, or system non-linearities that become more significant at low speeds. Further investigation is needed to determine whether additional data, higher sampling rates, or refined compensation algorithms could improve stability and consistency across the full range of operating speeds.

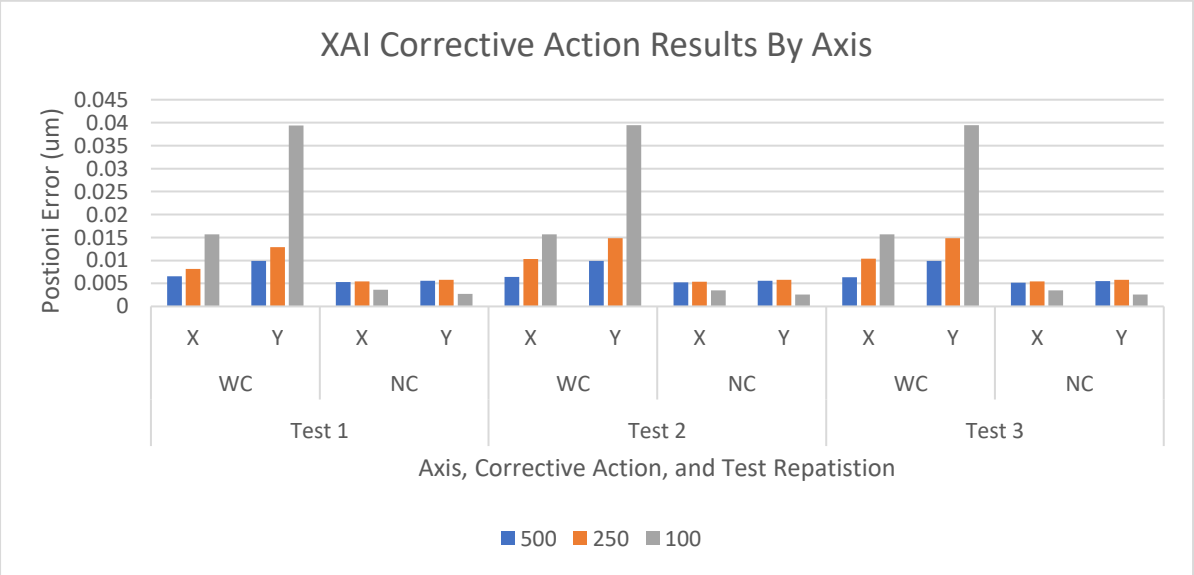


Figure 6-38 Hybrid mill XAI corrective action results by axis

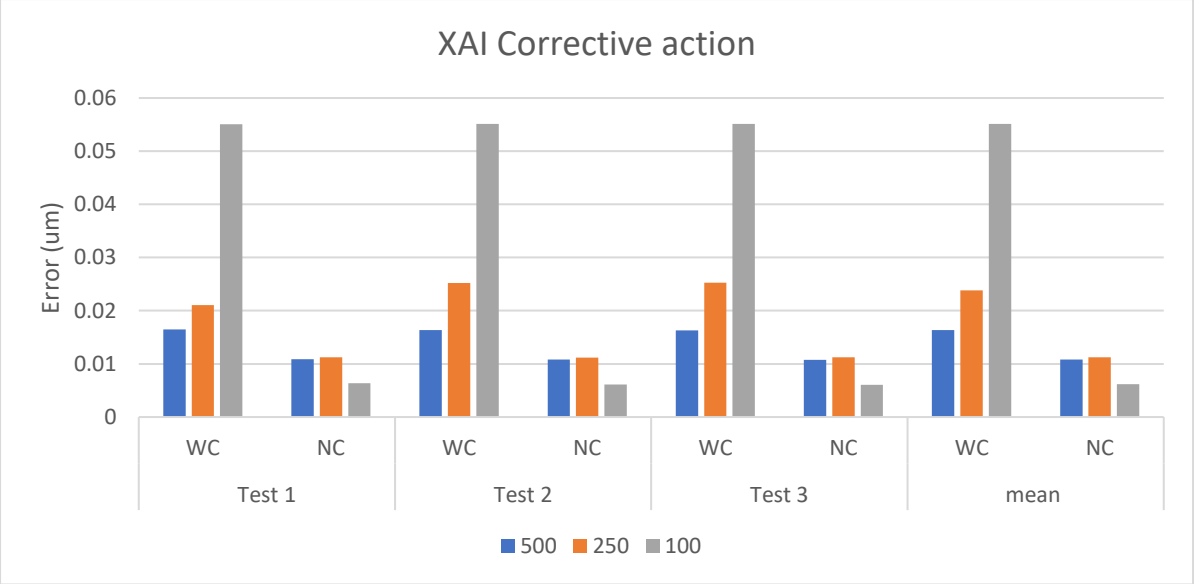


Figure 6-39 Hybrid mill XAI corrective action results combined

When reviewing the results from the XAI corrective action tests on the hybrid mill, the first notable observation is the difference in performance between the X and Y axes. The Y axis exhibited greater dynamic errors across all feed rates and tests, which aligns with expectations given its lower prediction accuracy, ranging from 44% to 71%, with an average of 56%. This limited reliability indicates that the Y-axis XAI prediction model struggled to accurately predict the dynamic error.

It is also observed that the smallest error reduction for the XAI occurred at the lower feed rates, with a maximum improvement of only −3.11%. This further supports the hypothesis that the system performs better at higher feed rates, possibly because the dynamic errors at low speeds are too small for the accelerometer to detect accurately. The reduced sensitivity of the accelerometers or insufficient capture of the hybrid mill's dynamic signatures could explain this behaviour.

Another key finding is that the spread of error for the XAI system is narrower compared to the RL model, suggesting that while XAI may not significantly reduce error, it produces more

consistent results. The cause of this consistency remains unclear and warrants further investigation.

Across both models, the best results occurred at the 500 mm/min feed rate. Interestingly, this is the same condition where validation accuracy was lowest (RL: 69–28% for X and Y; XAI: 86–71% for X and Y). This inconsistency suggests that a critical influencing variable may not have been captured during data collection. One possible missing factor is temperature. Since the experiments were conducted in an uncontrolled environment where ambient temperature varied between 16 and 21 °C, thermal expansion of steel components (10.8–12.5 µm/m °C) may have altered the hybrid mill's dynamic characteristics compared to those during training.

Another possible explanation is related to the frequency sensitivity of the accelerometers used. The current sensors, with a range of 0.5–3000 Hz, may be optimised for detecting high-frequency vibration peaks, whereas the hybrid mill's dynamic behaviour might be dominated by lower-frequency signals. Future tests will investigate this by using a low-frequency or higher-sensitivity accelerometer to determine if this improves model performance.

6.6 Summary

In this chapter, the development of a predictive digital twin-driven dynamic error compensation approach is established and evaluated on a COBOT and a hybrid mill.

For the COBOT, a predictive digital twin framework is developed using two accelerometers (wrist near TCP and base) together with encoder data and ball bar measurements as ground truth during training. Ten key acceleration features are extracted to support learning. A large training dataset is collected using ball bar tests across three radii, three feed rates, and three motion planes (XY, XZ, YZ), to collect relevant motion data within a work envelope. These data are processed and formatted into a single dataset for model training.

The QLattice XAI algorithm is used to generate a predictive equation for COBOT radial error using as few features as possible. The trained model achieved a high prediction performance of 99.4% and outperformed other tested machine learning methods, including KNN, decision trees, neural networks, random forest regression, and XGBoost.

A real-time corrective strategy is then tested, where predicted dynamic error is used to shift future COBOT target points before execution, reducing deviation during motion. Validation experiments on a semi-circular path at multiple feed rates show that applying this XAI-based compensation reduces dynamic error compared to uncompensated motion, demonstrating the feasibility of real-time predictive error compensation. However, current testing is limited to 2D (XY plane), and full 3D validation is identified as required future work.

The hybrid mill had RL and XAI models developed to predict the dynamic error. For the regression learning, the optimizable ensemble model produced the best performance with prediction accuracies of 88.3% and 71.6% for the X and Y axes, respectively. The XAI achieved 92% and 56% for the same axes. These models were subsequently tested using new ball bar experiments, where real-time corrective action was applied using an encoder offset method. This offset was controlled via an analogue voltage input to the A3200 controller, calibrated so that 1 V corresponded to 25 μm of positional adjustment, allowing a total correction range of ±125 μm. Importantly, this corrective method operated independently of the G-code and CNC trajectory planner, making it suitable for integration into existing systems.

During validation testing, both models were evaluated under identical conditions to the training phase. The results showed no consistent improvement in dynamic error reduction across all feed rates, with the exception of a marginal reduction at 500 mm/min using the RL model. Nevertheless, the consistency of the absolute mean error values across all repetitions indicates that the system was stable and repeatable.

The lack of measurable improvement is believed to be due to several possible factors. First, some key dynamic data may not have been captured due to limitations of the accelerometers used, suggesting that a lower-frequency or higher-sensitivity sensor could provide better results. Second, variations in the testing environment, particularly temperature and humidity, may have influenced machine behaviour, as the laboratory was not environmentally controlled. Finally, the resolution of the current sensors and control system may not be sufficient to achieve the sub-micron precision required for meaningful dynamic error correction.

This chapter has demonstrated that accelerometers have the potential to be used to measure and compensate for dynamic errors in the hybrid milling machine. The experimental results indicate that the proposed approach is capable of detecting machine dynamic behaviour outside the machine control loop. However, at the current stage of development, the system does not provide sufficient reliability for practical industrial implementation.

Consequently, the full completion of Research Question 3 has not been achieved, as a fully operational digital twin capable of applying direct real-time corrective action to the hybrid mill was not realised within the scope of this study. While the hybrid mill results were inconclusive in terms of reliable error compensation, the findings from the COBOT experiments and associated validation tests demonstrate that the proposed methodology is feasible and can successfully improve positioning accuracy in workpiece handling tasks.

The limitations observed in the hybrid mill implementation are primarily attributed to constraints within the experimental setup, including sensor resolution, environmental stability, and limitations in the available training data. These factors reduce the reliability of the predictive digital twin when operating at the higher precision levels required for ultra-precision machining.

Given these findings, further development of the predictive digital twin for the hybrid mill will be paused within the scope of this research. Future work should instead focus on establishing a more robust sensing and data acquisition framework before re-implementing the predictive correction system. Improvements such as higher-resolution sensors, improved environmental control, and expanded datasets for model training are expected to significantly enhance system reliability.

Therefore, while the hybrid mill digital twin could not be fully realised in this study, the research has established a foundational framework and demonstrated the feasibility of the proposed approach. Further investigation is recommended to develop the system into a fully operational digital twin capable of real-time dynamic error compensation in ultra-precision machining environments.

7 Chapter 7 –Digital twin-enabled automated dynamic error mitigation for COBOT in 3D space

7.1 Introduction

This section presents the continued development and testing of the digital twin-driven dynamic error compensation approach in the COBOT system, detailing its application and validation during full XYZ operations, as well as its integration for automated work handling tasks such as loading and unloading in a CNC mill.

The validation of the COBOT explainable AI (XAI) model introduces several research challenges for the digital twin framework. The first challenge is to determine the accuracy of the XAI model in predicting dynamic errors during real-time operation, particularly when the system is exposed to new motions and feed rates that were not included in the training dataset. This will assess the robustness and generalisation capability of the predictive model.

A second challenge is to evaluate the effectiveness of the digital twin in applying corrective actions within a three-dimensional workspace. This includes assessing how accurately the system can modify the target position to compensate for predicted dynamic errors during motion.

The final challenge is to evaluate the performance of the digital twin within a realistic pick-and-place scenario. This will determine whether the proposed approach is viable for practical tasks performed by collaborative robots and whether it can reliably improve positioning accuracy during automated handling operations.

The digital twin enhances the performance of the COBOT by predicting dynamic errors from acceleration signals through XAI-based modelling. This enables the system to apply corrective action outside of the conventional servo control loop, allowing for real-time compensation and

improved motion accuracy. Such advancements are essential for the automation of workpiece handling in ultra-precision manufacturing, where reduced setup time, improved repeatability, and the potential for lights-out manufacturing can significantly increase production efficiency.

The first part of this chapter focuses on testing the digital twin during XYZ motion of the COBOT. This research was presented at the 20th International Conference on Precision Engineering [98].

7.2 Methodology for testing the effectiveness of the digital twin-driven dynamic error compensation approach in XYZ motion

This section will cover how the digital twin operates while conducting movements in three dimensions, and the test that was conducted for its validation.

7.2.1 Operation of digital twin in XYZ

The digital twin was developed to enable real-time corrective action based on the predicted dynamic motion errors. This functionality is achieved using two accelerometers mounted on the COBOT: one positioned near the end effector to capture tool-side vibrations, and another attached to the COBOT base to measure ground vibrations and account for external interferences or random noise. The system employs an XAI model to predict dynamic motion errors from extracted accelerometer signal features.

The XAI model is implemented using *Qlattice*, a symbolic regression tool capable of generating interpretable mathematical relationships between sensor inputs and motion errors. The model is trained on historical data obtained from ball bar tests conducted in the XY, XZ, and YZ planes. This training process enables the digital twin to learn the dynamic characteristics of the COBOT and establish a predictive work envelope within which motion errors can be estimated.

Predicted errors are used to adjust subsequent target positions, ensuring the COBOT maintains trajectory accuracy through real-time compensations. Due to limitations in the COBOT's motion programming, corrective actions cannot be applied once a movement command has started. To address this constraint, the algorithm identifies the next two motion targets and applies corrections to them pre-emptively, reducing computational load and enabling faster update cycles.

A proof of concept for this approach has been demonstrated previously, showing improvements in COBOT trajectory tracking [94] and reductions in dynamic motion error within the XY plane [93]. In the present study, this capability is extended to full XYZ error monitoring and correction, further enhancing the robustness and applicability of the COBOT digital twin system.

The workflow of the COBOT corrective action system operates through the following sequence:

- The control program starts the COBOT movement.
- The algorithm identifies the subsequent motion targets in advance.
- Real-time acceleration signals are captured from the installed sensors.
- Key signal features are extracted from the accelerometer data for analysis.
- The XAI model uses the extracted features to calculate the predicted elastic deformation.
- New corrected target coordinates are computed based on the predicted deformation.
- The corrected target positions are sent to the COBOT controller for execution.
- The process repeats continuously until the completion of the programmed operation.

This process enables real-time prediction of COBOT dynamic error within a defined work volume, enhancing overall operational accuracy. This workflow is shown in Figure 7-1 below.

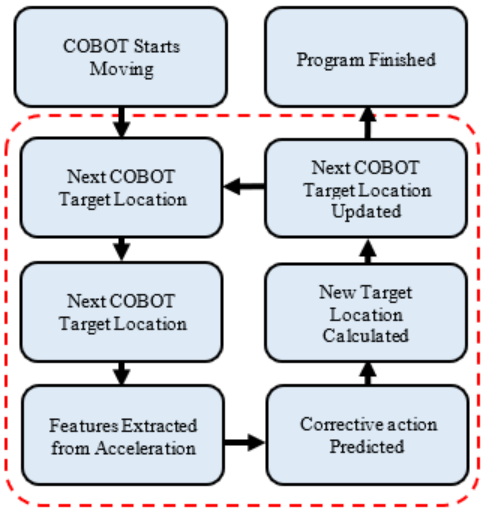


Figure 7-1 COBOT dynamic error corrective action work flow

Using this method, the corrective action of the COBOT is applied within the XY plane while the robot moves through three-dimensional space. This limitation exists because the XAI model was trained to apply corrections as radial deviations, rather than as independent corrections in the X, Y, and Z directions. Retraining the model for full three-dimensional corrective capability would require new training data obtained from a sensor capable of measuring the COBOT's position with micrometre-level accuracy across all three axes simultaneously. Without access to such a sensor, it is not currently possible to train the digital twin to operate effectively with true three-dimensional positional correction.

7.2.2 Experimental details for testing the proposed approach in three-dimensional motions

The experiments were conducted using a UR10e COBOT equipped with a Robotiq HandE end effector for workpiece handling. Acceleration data was captured using PCB model 356B18 tri-axial accelerometers, connected through a National Instruments cDAQ-9174 system with NI-9234 modules. Dynamic motion errors were measured using a Renishaw QC20-W ball bar. The system operation and data handling were facilitated through RoboDK v5.5.3 for COBOT motion control, MATLAB 2023a for digital twin modelling, data communication, and overall system coordination, and Ball Bar Trace 1.2 SP2 for real-time ball bar data acquisition.

The COBOT was fitted with an attachment held by the HandE end effector to mount the accelerometer as close as possible to the TCP to ensure accurate measurement of the motion dynamics, as shown in Figure 7-2(b). The COBOT's motion began from a position located 150 mm from the centre of the ball bar mount, executing a programmed trajectory consisting of three circular paths at varying heights, with transitions between each circle, as illustrated in Figure 7-2(a). In this study, only one radius was tested. Since ball bars are industry-standard measurement tools in precision manufacturing, the recorded data provides a highly reliable representation of the true TCP position. The complete motion sequence, including circular paths and transitions, covered a total distance of 3485.264 mm.

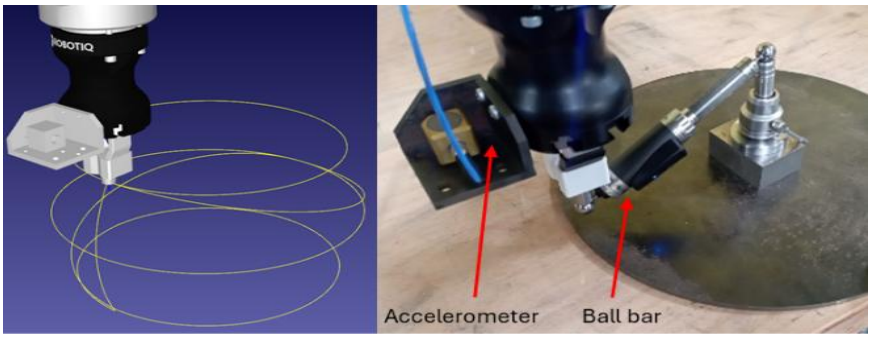


Figure 7-2 COBOT sensor and control software (a) RoboDK COBOT path (b) Set up of ball bar and accelerometer.

The three circular paths and the transition movements between them are illustrated in Figure 7-3, which also shows the corresponding circle dimensions and height differences. In this experiment, the COBOT's motion will be executed only in the clockwise direction to maintain consistency across all tests and reduce potential variability from directional effects.

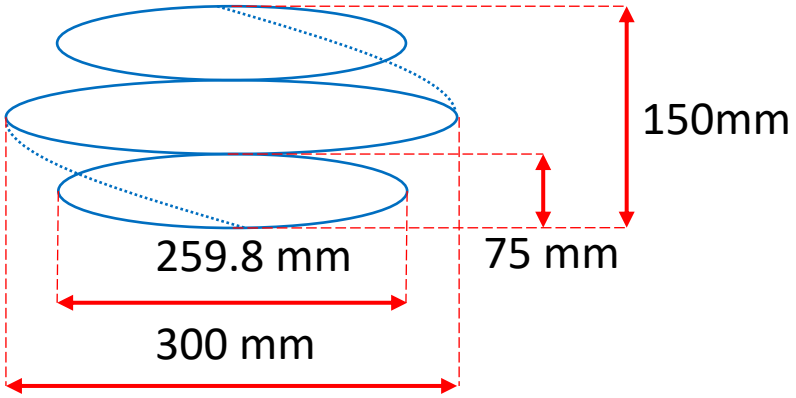


Figure 7-3 COBOT XYZ tool path dimensions

Corrective action was implemented by predicting dynamic motion errors derived from key features extracted from the accelerometer signals. The same prediction equation, represented by equation 6-1, was employed to calculate the radial corrections (RC) applied to the X and Y axes of the COBOT.

The key accelerometer variables identified by the XAI model are F1 = Peak Amplitude (X-axis), F2 = Band Power (Y-axis), F3 = Root Mean Square (X-axis), and F4 = Kurtosis (X-axis). The machine learning technique used to generate the prediction equation 6-1 was previously presented by Abhilash et al. [93] and discussed in detail in Chapter 3. The XAI employs symbolic regression, chosen for its transparency, computational efficiency, and predictive capabilities.

Once COBOT motion commences, the accelerometer signal features are extracted to predict dynamic errors using equation 6-1. The resulting corrective actions are applied to the X and Y positions of future targets. This prediction-and-correction loop continues in real time until the COBOT motion concludes. Experiments were conducted at speeds of 1000, 3000, and 6000 mm/min, with three repetitions for each condition. The test parameters are summarised in Table 7-1.

Table 7-1 COBOT XYZ XAI validation test conditions

RADIUS (MM)	SPEED (MM/MIN)	WITH COMPENSATION (WC)			NO COMPENSATION (NC)		
150	1000	Test 1	Test 2	Test 3	Test 1	Test 2	Test 3
150	3000	Test 1	Test 2	Test 3	Test 1	Test 2	Test 3
150	6000	Test 1	Test 2	Test 3	Test 1	Test 2	Test 3

The parameters for the XYZ toolpath were carefully chosen to capture the dynamic error signatures of the COBOT, enabling the XAI to generate accurate predictive equations in real time. A radius of 150 mm was selected instead of the 100 mm default of the ball bar, providing a larger test volume for more comprehensive data collection. Due to calibration equipment limitations, radii outside the training data range could not be tested.

The selected speeds of 1000, 3000, and 6000 mm/min span a wide range of typical COBOT operational speeds. Including 1000 and 3000 mm/min also allows assessment of the digital twin's performance at feed rates not explicitly covered in the training data. To ensure data reliability, each test is repeated three times, providing robust information for the XAI model.

For validation, the same motion parameters are applied to evaluate the digital twin's effectiveness under With Compensation (WC) and No Compensation (NC) conditions. This produces nine validation results for each scenario, yielding a total of eighteen results, ensuring a thorough evaluation of the digital twin's predictive and corrective capabilities.

7.2.3 Data processing

The results of this experiment will be collected using Renishaw Ball Bar Trace 1.2 software, which enables real-time monitoring of ball bar readings. This software change was necessary because the Renishaw Ball Bar 20 could not capture sufficient data for the complete motion in one reading.

Using Ball Bar Trace 1.2, the ball bar can be connected directly to the software, providing a real-time display of the collected data, which facilitates continuous observation and ensures accurate data capture throughout the motion shown in Figure 7-4.

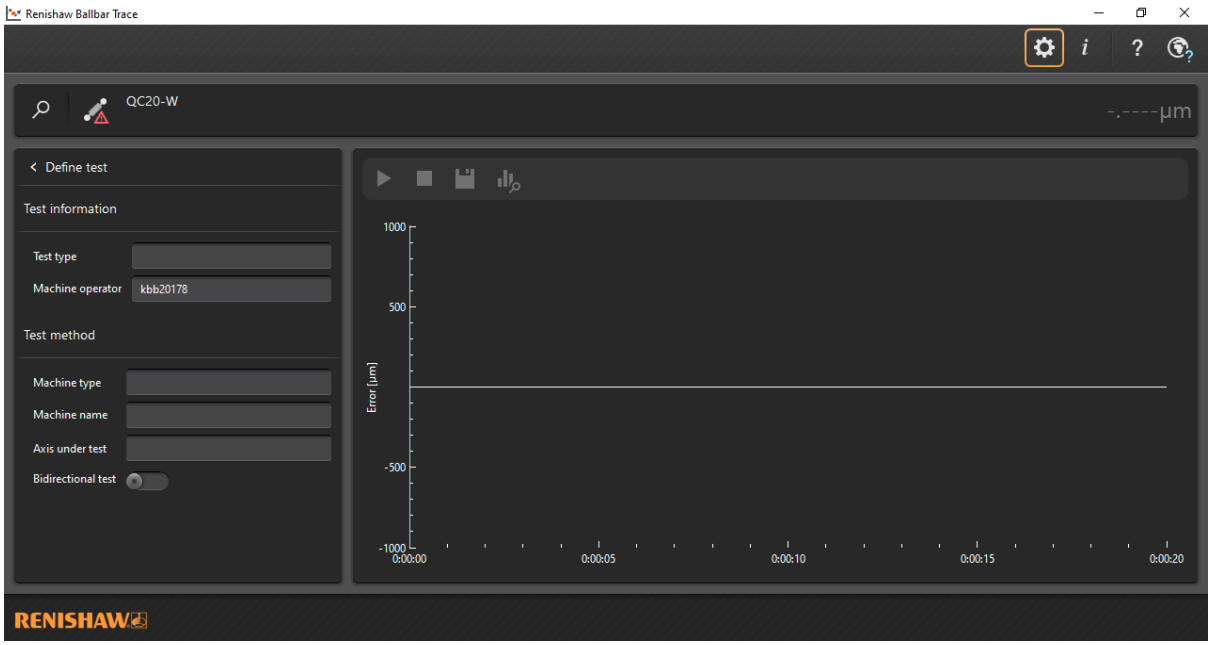


Figure 7-4 Ball Bar Trace software

Once the data is collected, it can be saved in the Ball Bar file format. These files can then be opened in CARTO-Explore for review, as shown in Figure 7-5. Alternatively, the files can be exported as a Comma-Separated Values (CSV) by changing the file type, since the structure of the data is compatible with standard spreadsheet formats, as illustrated in Figure 7-6.

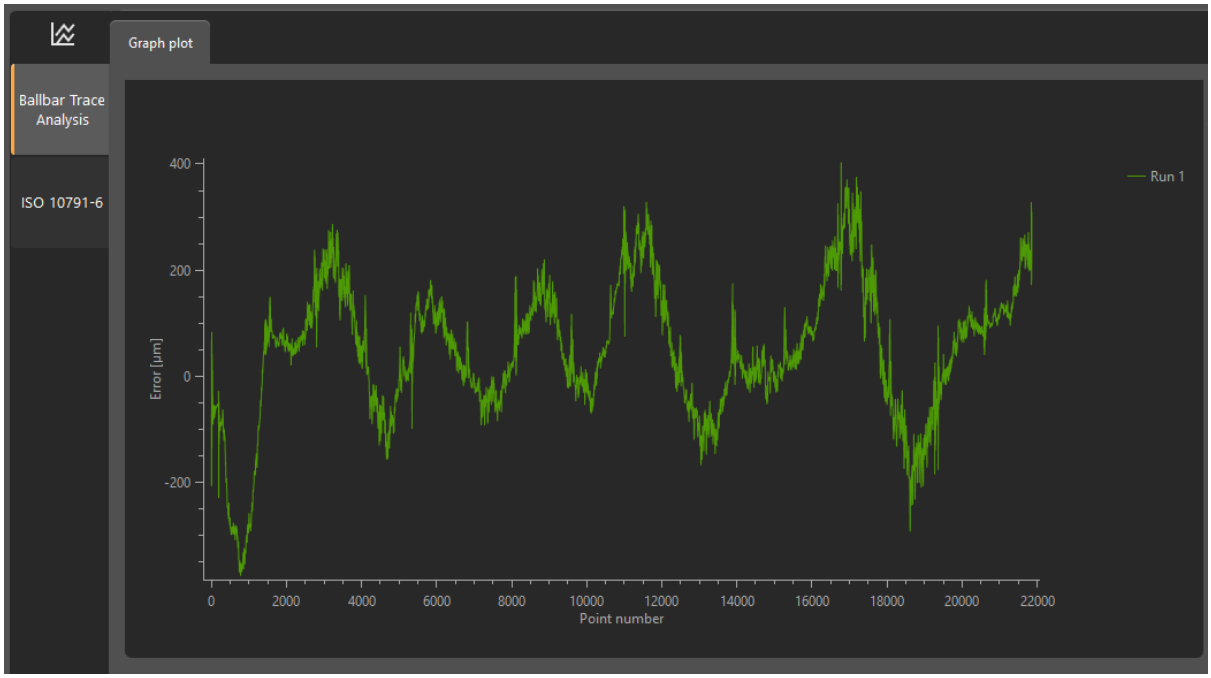


Figure 7-5 CARTO-Explore reviewing ball bar data


```
{"AxisUnderTest":"XYZ","BallbarCalibrationCertificate":"3RGK88-211011-00",
"BallbarCalibrationDate":"0001-01-01T00:00:00","BallbarCalibrationOrganization":"Renishaw plc",
"BallbarManufacturer":"Renishaw","BallbarModel":"QC20-W","BallbarName":"3RGK88",
"BallbarSerial":"3RGK88","ErrorUnits":"millimetres","IsBidirectional":false,"Machine":"UR10e",
"MachineType":"Cobot","Operator":"kbb20178","Runs":[{"Direction":0,"Readings":[-0.070085994597527415,
-0.16441383149986355,-0.2067611165527472,-0.14769036219597698,-0.050356997879948871,-0.
0036803295367837858,0.065873589905034224,0.083725557819604737,0.063611670632091138,0.
025347486077414817,-0.041071778009582526,-0.10758133733728593,-0.13554303310914623,-0.
10745922026389744,-0.037162025410193582,0.0231472836954195,0.042216773883282356,0.033415171088174704,0.
0005970929866831278,-0.042293553123897938,-0.070574601878645657,-0.0840105991783066,-0.
084499155184108957,-0.064283647119434281,-0.029281097507804683,0.0034691472303710995,0.
011596755083610213,-0.010829453563611553,-0.042721171868869388,-0.069780614141270722,-0.
088041131866709046,-0.08462129390195576,-0.069047698205973049,-0.055732364694930307,-0.
```


Figure 7-6 Ball Bar Trace data file format

This method of converting the data into a usable format allows it to be easily loaded into MATLAB, enabling direct comparison between the NC tests and the WC tests. Once the data

is extracted as a CSV, it can be loaded into other software. Figure 7-7 shows the data presented in Excel based on the date exported from CARTO-Explore.

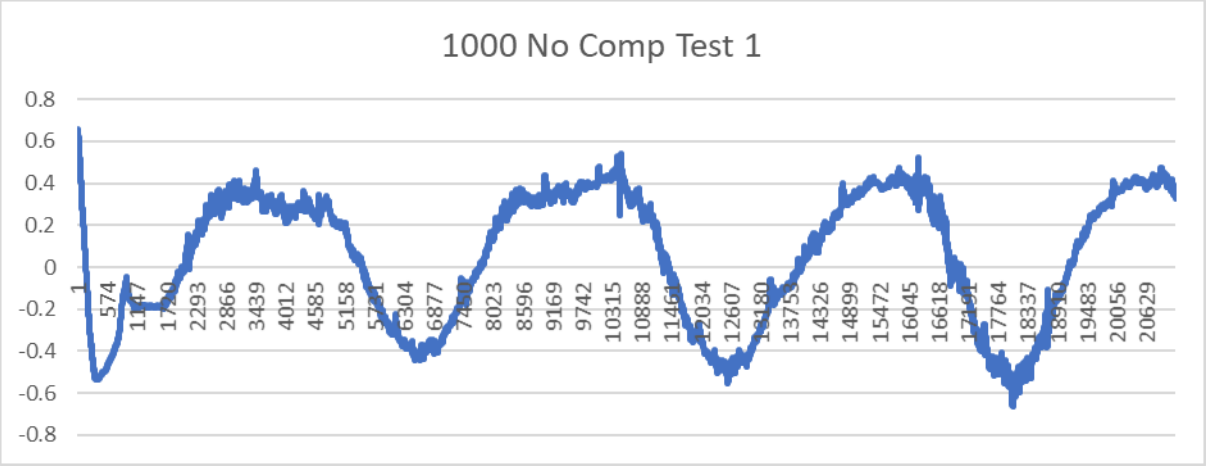


Figure 7-7 Comparing results in Excel from CARTO

To collect the data, the Ball Bar Trace software requires an in-feed and out-feed, similar to the Renishaw Ball Bar 20 software. The in-feed portion is visible in the data and must be removed, as it contains random errors unrelated to the actual motion or corrective actions being tested. To address this, a predetermined amount of data is removed from the start of each test. The optimal amounts to remove were determined based on test results: 1000 samples for 1000 mm/min, 300 samples for 3000 mm/min, and 90 samples for 6000 mm/min.

Since all data is collected along the same tool path, the results can be directly compared without additional filtering or processing. This approach allows for a straightforward comparison between NC and WC, with the three repetitions for each feed rate averaged to determine the overall performance.

7.2.4 Results and discussion

The results of the experiments are presented in Figure 7-8 and summarised in Table 7-2. The data is divided into two groups: NC, shown in blue, and WC, shown in orange. From Figure 7-8, it is evident that applying compensation significantly reduces the Mean Absolute Error

(MAE) across all feed rates and repetitions. Table 7-2 provides a quantitative overview, showing that the use of the proposed dynamic error compensation approach reduces the MAE by 51% to 61% and decreases the standard deviation by 57% to 64%, demonstrating the effectiveness of the corrective action.

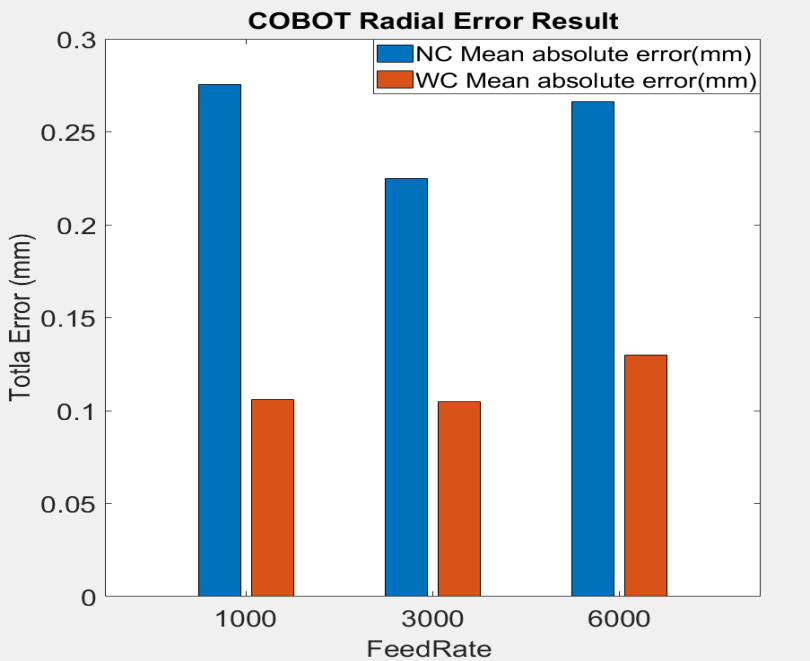


Figure 7-8 COBOT Radial Error Results

Table 7-2 Dynamic error testing result of the COBOT

	Mean error Test 1 (mm)	**Mean error Test 2 (mm)**	**Mean error Test 3 (mm)**	**Mean Error All Tests (mm)**	**Error Reduction Mean Error (%)**	**All Test Mean Standard deviation**	**Standard deviation reduction (%)**
1000 NC	0.2805	0.2758	0.2696	0.2753	61.57	0.2974	64.66
1000 WC	0.1052	0.106	0.1061	0.1058		0.1051	
3000 NC	0.1626	0.2475	0.2642	0.2247	53.36	0.2647	56.1
3000 WC	0.1206	0.0952	0.0985	0.1048		0.1162	
6000 NC	0.2711	0.2712	0.2562	0.2662	51.13	0.2887	57.4
6000 WC	0.1298	0.13	0.1304	0.1301		0.123	

Figure 7-8 shows the NC and WC results as the MAE for each feed rate tested. The results clearly demonstrate a reduction in radial error when using the proposed dynamic error compensation approach. MAE was chosen as the error metric because it accounts for both positive and negative deviations, reflecting the TCP displacement relative to the 150 mm radius of the ball bar.

The error bars in Figure 7-8, along with the low standard deviation values, are reported in Table 7-2, indicating an increase in the COBOT's stability during motion. Specifically, the standard deviation was reduced by 56% to 64%, resulting in an average reduction of 59.39%. Numerically, the proposed approach achieved an overall error reduction of 51% to 61%, with an average reduction of 55.35%, as summarised in Table 7-2.

It should be noted that an anomaly was observed in Test 1 at 3000 mm/min, which appeared only once and did not significantly impact the overall results. This data point should be considered an outlier, and further investigation is warranted to identify its cause and potential solutions.

As anticipated, the reduction in error decreases with an increase in feed rate, reflecting the proportional relationship between dynamic error and travel speed. Nonetheless, the proposed approach successfully reduces both the mean error and the standard deviation across all tests, demonstrating significant improvements in the COBOT's accuracy and precision.

Importantly, these tests involved movements different from the training data, including feed rates of 1000 mm/min and 3000 mm/min, which were not part of the training set. The minimal MAE difference of just 1 μm between these feed rates indicates that the COBOT maintains similar precision even at three times the training speed. Additionally, the reduction in standard deviation (ranging from 56.1% to 64.66%) highlights a notable improvement in the repeatability and reliability of the COBOT's motion along the tool path.

The proposed digital twin, leveraging accelerometer signals, provides a cost-effective solution to enhance COBOT precision and accuracy through real-time corrective actions. Future studies should explore its performance in non-circular motions, operations beyond the trained speed range, and varying orientations of the TCP, to assess the impact of gravity and joint loading on system performance.

While the proposed approach has demonstrated considerable promise in reducing dynamic errors in real time, additional research is necessary to fully understand its limitations and refine its capabilities for broader industrial applications.

7.2.5 Summary of the test

This chapter presents the development of a digital twin aimed at enhancing automated workpiece handling in ultra-precision manufacturing. The primary objective is to reduce the dynamic error of a COBOT during real-time operations using accelerometer measurements to monitor displacement.

The experiment evaluated a digital twin-driven system that predicts and corrects dynamic errors in real time using data from two accelerometers processed by an XAI model. Validation involved testing untrained XYZ motions both with and without digital twin compensation across multiple feed rates. The results demonstrated that the digital twin reduced radial error by an average of 55.35% and decreased standard deviation by 59.39%, improving both the accuracy and reliability of the COBOT. These findings indicate that the proposed approach enables higher-speed operations with minimal loss of precision, offering a cost-effective solution for ultra-precision manufacturing.

The system was designed to integrate seamlessly with existing workflows, requiring no modifications to the original toolpath, thereby facilitating easy implementation. Future research should investigate the limitations of the digital twin, particularly its performance across different COBOT joint orientations. Since the XAI model was trained for a specific orientation, its effectiveness may decrease in alternative orientations. Evaluating performance according to ISO 9283:1998, which provides comprehensive performance criteria and test methods for industrial robots, would allow for an objective assessment of the digital twin across varying orientations and operational conditions. Additionally, the influence of material weight and

workpiece security on COBOT performance was not considered in this study, but may introduce additional errors in real-world applications and warrants further investigation.

7.3 COBOT pick and place test

Based on the previous results, a final experiment is planned to evaluate the effectiveness of the COBOT's digital twin in real-world workpiece handling applications. The objective of this experiment is to quantify the COBOT's accuracy and precision when using the digital twin-driven dynamic error compensation approach compared to its default control.

Specifically, the test will measure how accurately and precisely the COBOT can place material in the desired location during automated work loading on a CNC mill machine. The results will demonstrate the potential of the proposed approach to enable reliable, automated handling for ultra-precision CNC machining operations.

7.3.1 Test methodology

This experiment evaluates the effectiveness of the proposed dynamic error compensation approach in COBOT's real-world workpiece handling tasks by measuring its accuracy and precision compared to the default control. The study involves the repeated placement of a tungsten carbide slip gauge block onto a CNC mill, with the block's location measured using a probe. This allows for quantification of placement error and comparison between WC and NC scenarios.

The selected placement location was chosen to assess the accuracy and precision of the COBOT digital twin in Cartesian space. The orientation of the gauge block will not be assessed in this experiment, as the proposed approach currently does not have the capability to measure or correct rotational errors. Since the COBOT's motion in this experiment is primarily translational, placement will be executed using MoveL and MoveJ commands:

MoveL (Move Linear): This command instructs the robot to move its end-effector along a straight line in Cartesian space from its current position to a specified target.

MoveJ (Move Joint): This command moves the robot's joints from their current positions to the specified target positions in joint space, following the most efficient or fastest path, which is generally not linear in Cartesian space.

The placement and probing process will be conducted at feed rates of 1000, 3000, and 6000 mm/min, using both MoveL and MoveJ commands, under WC and NC conditions. Each test will be repeated ten times, resulting in a total of 120 tests. This high number of repetitions ensures that the repeatability and reliability of the digital twin can be thoroughly evaluated. Since dynamic errors are predicted by the digital twin in real time, multiple repetitions help capture potential variations in performance and verify system robustness.

The process of the experiment will be as follows:

- The COBOT moves to the gauge block and picks it up using the electromagnetic attachment.
- The COBOT places the block onto the electromagnetic fixture on the mill bed.
- With the block secured, the CNC mill probes the XYZ location of the block to verify its position.
- Probing is performed at two points along the XY plane to also determine the angular orientation; the same points are probed consistently across tests.
- The measured XYZ location is compared with the intended target position to determine placement accuracy.
- This process is repeated ten times at feed rates of 1000, 3000, and 6000 mm/min, both with and without the digital twin active.

- After probing, the block is returned to the starting jig and positioned against a reference wall to ensure consistent initial placement for each repetition.
- The same target locations are used for all tests to maintain comparability.
- Experiments are conducted under both WC and NC conditions.
- The entire procedure is repeated using both MoveL and MoveJ motion commands.

The location of the gauge blocks will be calculated based on the probe readings. Before the experiment, the probe is calibrated to the X0 Y0 point above the hybrid mill's electromagnetic fixture. As two electromagnets are used, there will be a period during the transfer of the gauge block when both electromagnets are energised simultaneously while the gauge block is positioned between them. This approach prevents sudden changes in magnetic force that could cause the gauge block to move and alter its placement. This point serves as the origin of the experiment and represents the intended location to which the centre of the gauge block should be returned.

The centre of the block is determined by probing two known points along both the X and Y sides of the block. Using these four points in conjunction with the known dimensions of the gauge block, the central position can be accurately calculated. This method also allows the orientation of the gauge block to be assessed, enabling the calculation of the X and Y angles to account for any rotational errors during placement. The concept of this probing method can be seen in Figure 7-9 below.

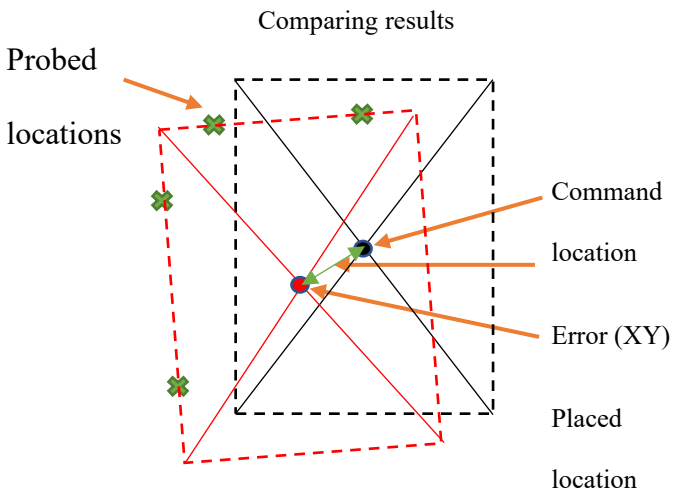


Figure 7-9 COBOT probing concept

A gauge block was selected for this experiment due to its high-precision manufacturing, which minimises dimensional errors, and its steel construction, which allows it to be manipulated using an electromagnet. The specific block used is a Rubert & Co. metric slip gauge, number 12035, with a nominal size of 20 mm.

During the experiment, the gauge block is expected to experience a slight temperature rise due to the use of the electromagnet, which can induce eddy currents, hysteresis losses, and resistive heating. Before probing, the temperature of the gauge block was measured using a Type 12035 thermocouple to allow for thermal compensation. The manufacturer specifies the thermal expansion of the gauge block as 5–6 µm/°C; for this experiment, a value of 5 µm/°C is used for calculations.

The nominal dimensions of the gauge block are 35mm×20mm×0.6mm (length, width and thickness), which will be used to calculate its centre and account for thermal expansion.

The experimental setup of the COBOT and hybrid mill is shown in Figure 7-10. The figure illustrates the positions of the jig and the electromagnets used for both the COBOT and the hybrid mill. The COBOT will pick up the gauge block from the jig located on the mill table

using its electromagnet and will follow the programmed tool path to place it onto the hybrid mill electromagnet. This will secure the gauge block in position for probing by the mill.

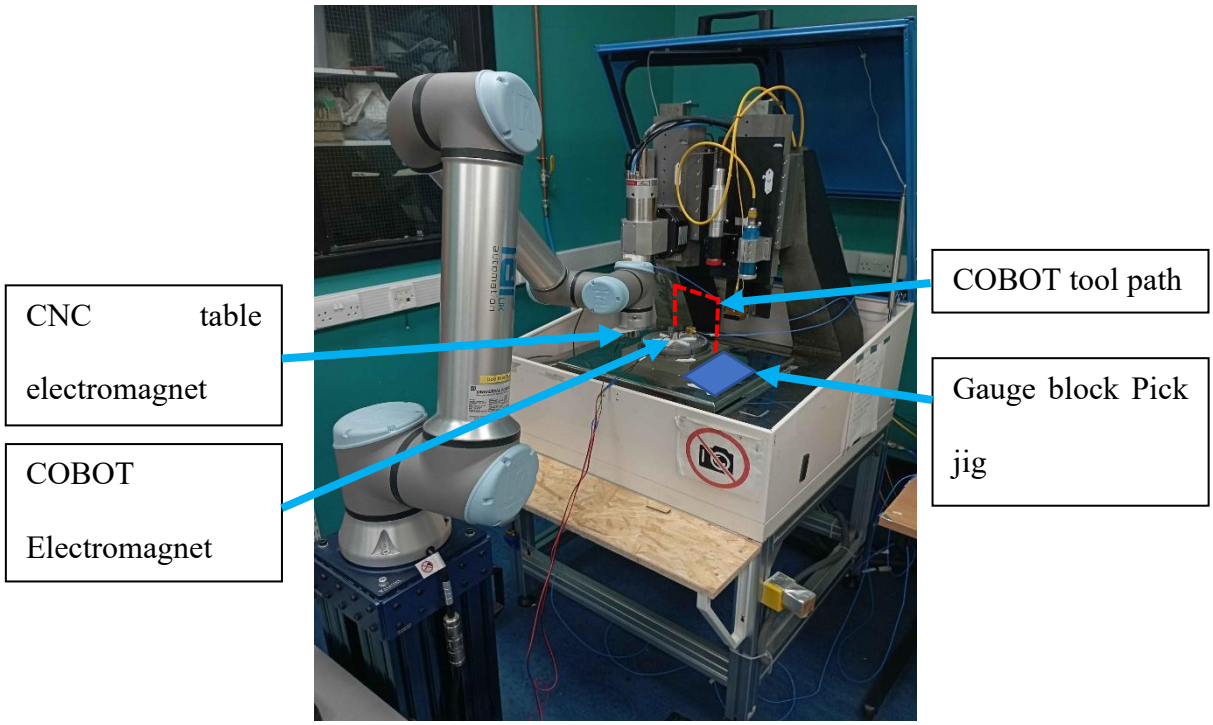


Figure 7-10 Experimental setup for pick and place test

All attachments for this experiment were 3D-printed using PLA. This approach was chosen to minimise errors arising from the design or machining of jigs and mounts. All attachments were securely bolted to their respective surfaces, with the exception of the jig, which was fixed in place using double-sided tape.

7.3.1.1 MoveL and MoveJ movement paths

As previously stated, two different movement commands will be employed to evaluate the effectiveness of the corrective action when the COBOT operates using different motion types, as both the training data and validation tests utilised MoveC commands. Figure 7-111 illustrates the tool path for the pick-and-place operation using MoveL, while Figure 7-12 shows the corresponding tool path for MoveJ. The same target locations are used for both movement commands, ensuring that the tool paths pass through identical points in each test.

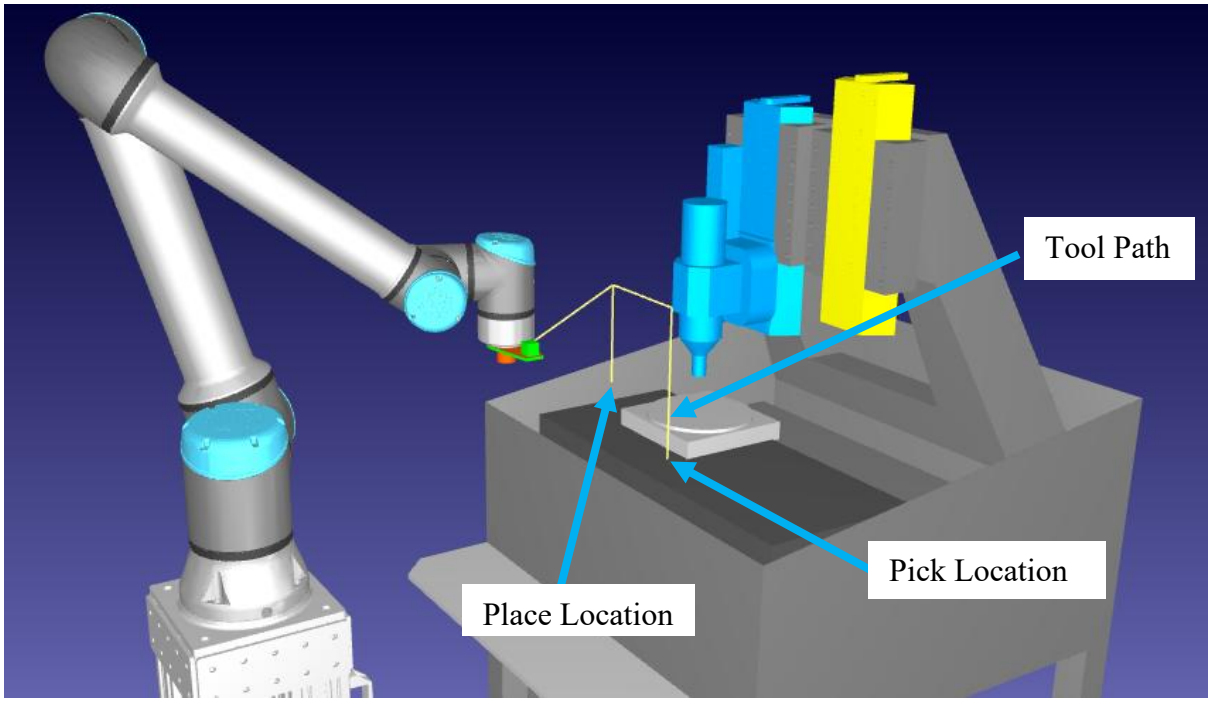


Figure 7-11 Move L pick and place tool path

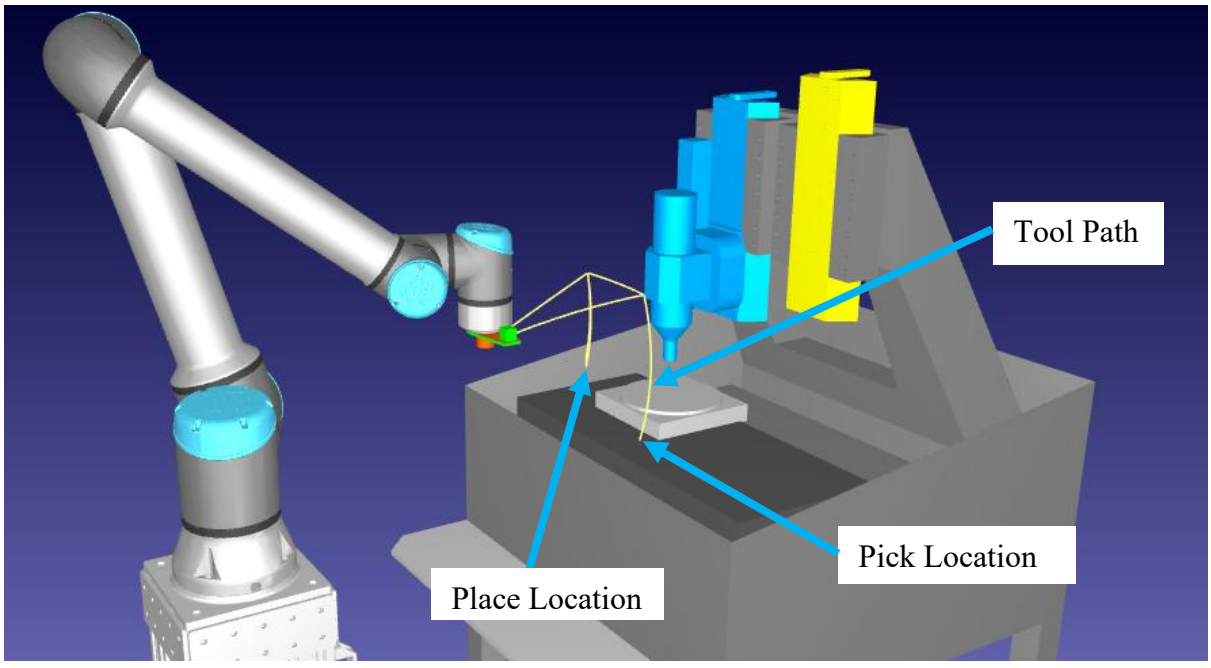


Figure 7-12 Move J pick and place tool path

7.3.1.2 Designs of mounts and jigs

7.3.1.2.1 Jig design

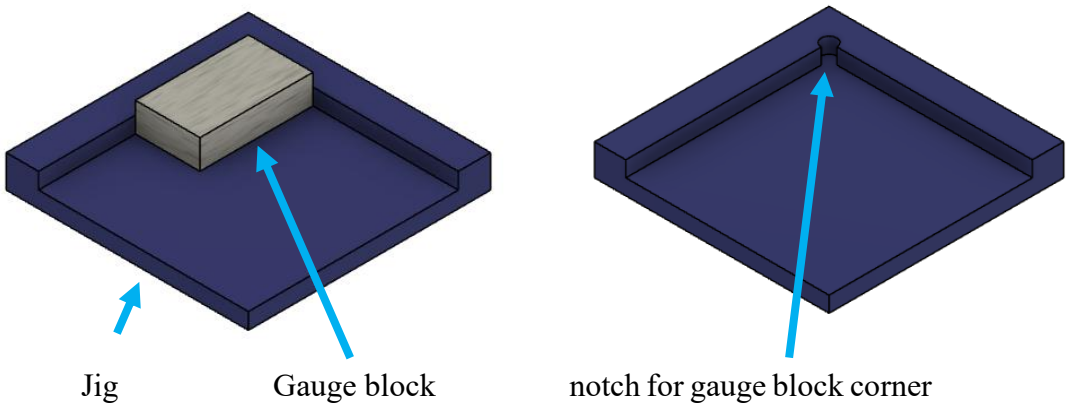


Figure 7-13 Design of gauge block locator

The designed jig is shown in Figure 7-13. The design allows for quick and repeatable resetting of the gauge block for subsequent tests. To address the limitations of 3D printing, particularly the difficulty in achieving perfect 90° angles without post-processing, a notch was incorporated into the corner of the jig. This feature ensures that the gauge block fits squarely and consistently during each test.

Any errors arising from the printing process, gauge block placement, or COBOT calibration relative to the intended location will be consistent across all tests. These systematic errors can therefore be accounted for by averaging the measured deviations across all repetitions and applying a corresponding offset. To maintain consistency, the same home location will be used throughout testing, and the COBOT, jig, and mounts will remain fixed until all tests are completed.

7.3.1.2.2 Hybrid mill electromagnet workpiece mount design

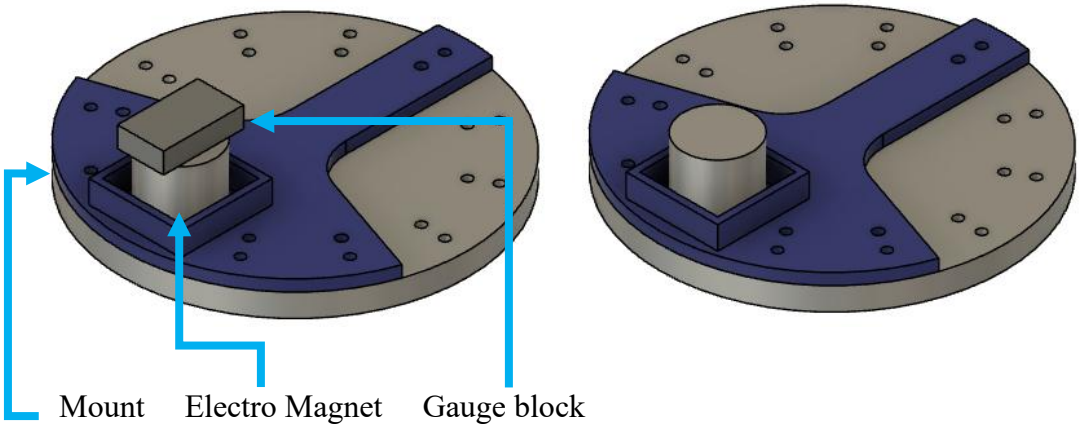


Figure 7-14 Design of electromagnet mount

The designed workpiece mount for the hybrid mill is shown in Figure 7-14. The design allows for easy attachment to the C-rotational table of the mill while also facilitating accurate measurement of the orientation and location of the electromagnet. The electromagnet is secured to the 3D-printed mount from the underside using a countersunk bolt. The surrounding wall structure enables the centre of the electromagnet to be measured by probing; as the centre of the bolt hole is known relative to the exterior of the wall, the X and Y coordinates of the centre can be determined.

This mounting method will introduce a small error due to the inherent flatness imperfections of the 3D-printed wall structure. However, this error is systematic and consistent across all tests. Consequently, it can be accounted for by averaging the measured deviations across all programmed movements and subtracting this offset from the results.

The centre coordinates of the electromagnet, as determined using the wall structure and probing, were:

X114.982553 Y4.107348 Z46.041826

These global coordinates were used as the origin for all experiments, as the centre of the electromagnet remains fixed across tests. The Z-coordinate was established by placing the gauge block on the hybrid mill electromagnet while powered and probing the height.

7.3.1.2.3 COBOT electromagnet sensor mount design

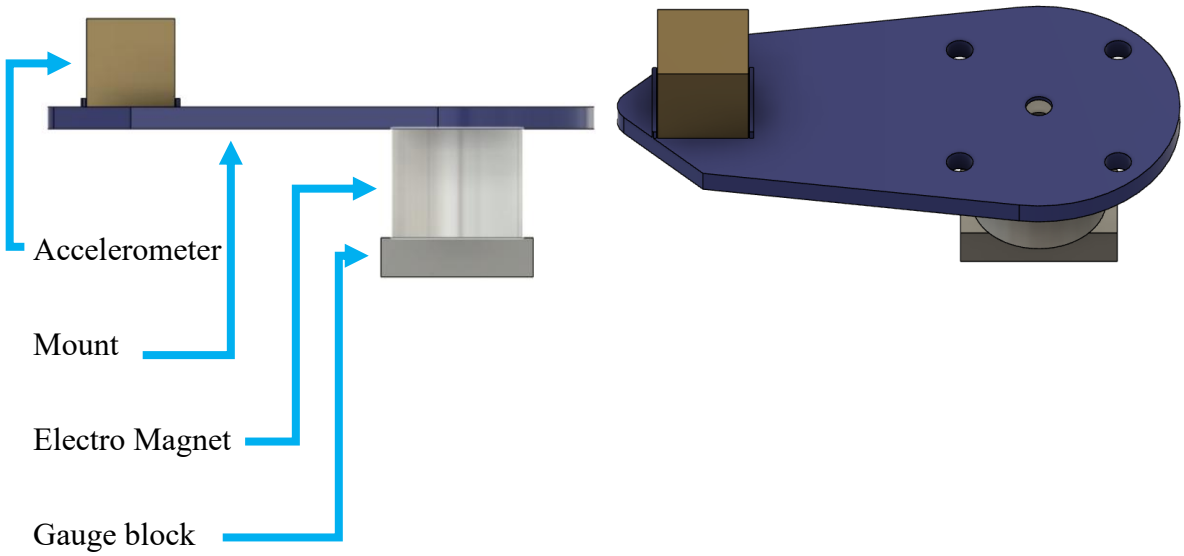


Figure 7-15 Schematics of the accelerometer and its electromagnet mount

The mounting for the accelerometer on COBOT is illustrated in Figure 7-15. The sensor mount is designed to securely hold both an accelerometer and an electromagnet. Both components are bolted to the mount to ensure stable attachment during operation. The mount itself is fixed to the COBOT at precise angles to facilitate straightforward calibration of the robot's axes. It was 3D-printed to allow for rapid development and to incorporate walls that accurately align the accelerometer at the correct orientation.

7.3.2 Equipment used

The experiments were conducted using a UR10e COBOT equipped with a custom electromagnet mount and a 24 V, 80 N electromagnet. Acceleration data were captured using PCB model 356B18 tri-axial accelerometers, interfaced through a National Instruments cDAQ-9174 system with NI-9234 modules. The location of the gauge block was determined using a

Hexagon M&H infrared probe (IRP40.50). RoboDK v5.5.3 was used for COBOT programming and operations, MATLAB 2023a for digital twin modelling, data communication, and comprehensive system control, and Microsoft Excel for post-processing and analysis of results. The Aerotech A3200 software served as the control platform for the hybrid mill.

7.3.3 Program operation and data collection

The COBOT control and data acquisition are managed using MATLAB, similar to the previous training and validation experiments, except for the Aerotech A3200 CNC controller, which records the probe XYZ locations. The MATLAB program differs slightly from the previous implementation in that the corrective action is applied only to the placement location, ensuring that its coordinates are continuously updated, as it is the sole target requiring adjustment.

During testing, MATLAB collects acceleration data in the X, Y, and Z directions from the COBOT wrist and base, along with the predicted corrective actions and the updated placement coordinates. Simultaneously, the Aerotech A3200 collects and displays the probe location for each test point. Once displayed, these probe measurements are saved and subsequently used for result calculations.

The COBOT motion is programmed in RoboDK, with seven target positions defined. The sequence of targets, along with their corresponding designations and functions, is summarised in Table 7-3. The target locations for MoveL commands are illustrated in Figure 7-16. With all targets remaining the same as shown in Figure 7-17 for both tool paths.

Table 7-3 COBOT pick and place the target order

Program steps	Target Name	Description
1	Move away	This moves the COBOT above the pick-up location
2	Pick	This is the pick location and is set up to touch the gauge block for the electromagnet to attach to it.
3	Move away	This moves the gauge block up to begin the placement process
4	Start Place	This location is above the place location to allow for a slight line to place the gauge block.
5	Place Mid-Point	This is a point for a slight interruption to allow for the corrective action to be applied to the place location.
6	Place	This is the point where the gauge block will be in contact with the electromagnet on the CNC base.
7	Start Place	This retunes the COBOT to a point safely above the place location
8	Safe Away	To bring the COBOT to a safe location where it will not interfere with the motion of the CNC while probing the block

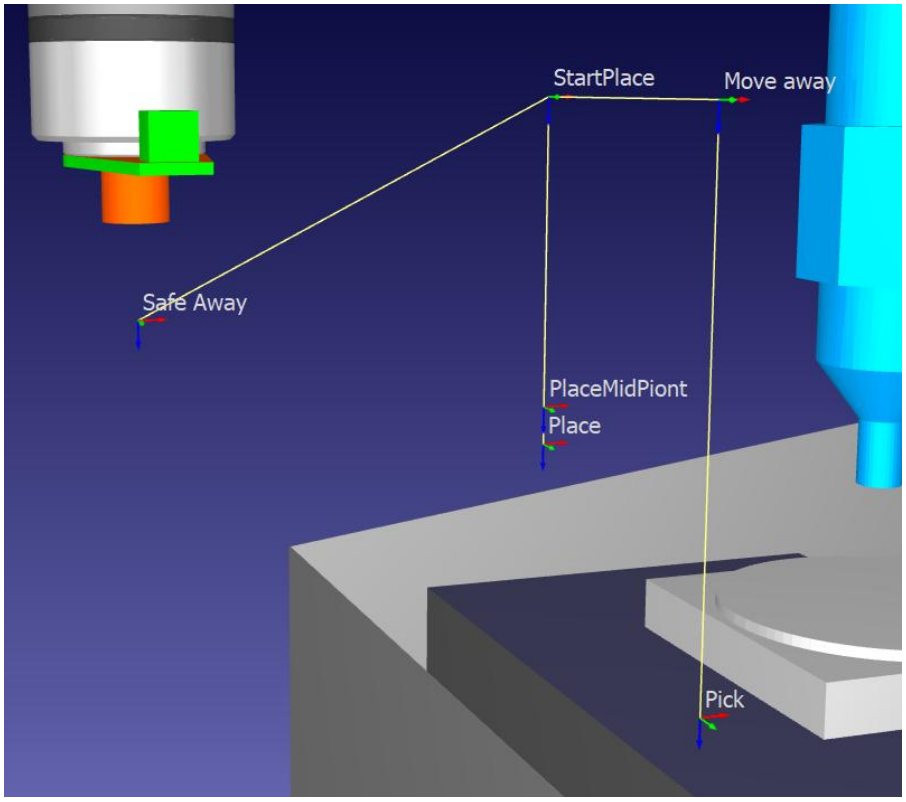


Figure 7-16 COBOT target locations in RoboDK

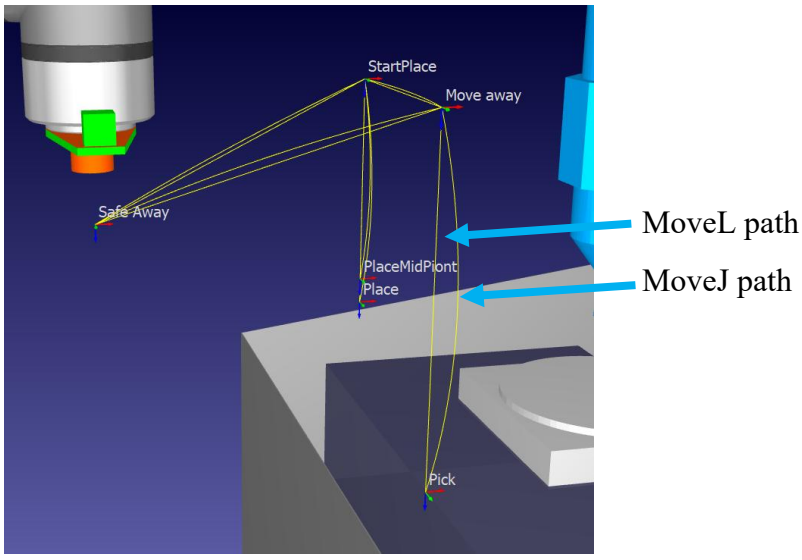


Figure 7-17 COBOT target locations matching for Move L and J toolpaths

During the placement of the gauge block onto the hybrid mill electromagnet, a short delay is introduced to allow the electromagnets to reach full magnetic strength. This prevents the gauge block from shifting during the placement stage. During this delay, the COBOT is stationary, and therefore, the dynamic error caused by motion is effectively eliminated. As the corrective

action for predicted placement error is applied to the target position before placement, no additional adjustment is required while the COBOT remains stationary. This approach is valid for pick-and-place operations where the robot pauses during placement. However, it would not apply to tasks involving continuous motion or other operations where the robot does not stop during the task. Once the gauge block has been placed on the electromagnet on the CNC milling machine base and securely fixed, the automated probing routine is initiated. This probing code has been developed to enable fast and repeatable measurement of the gauge block at predefined locations. The probing points have been selected to capture key dimensions: two points on the X-axis (X1 and X2), two points on the Y-axis (Y1 and Y2), and one point on the top surface along the Z-axis. The probe coordinates are referenced relative to the centre of the gauge block's top surface and are summarised in Table 7-4. The path followed by the probe during measurement is illustrated in Figure 7-18.

Table 7-4 gauge block probe locations

Target Name	X location	Y location	Z location
X1	10	-10	-4.5
X2	10	10	-4.5
Y1	7.5	17.5	-4.5
Y2	-7.5	17.5	-4.5
Z	0	0	0

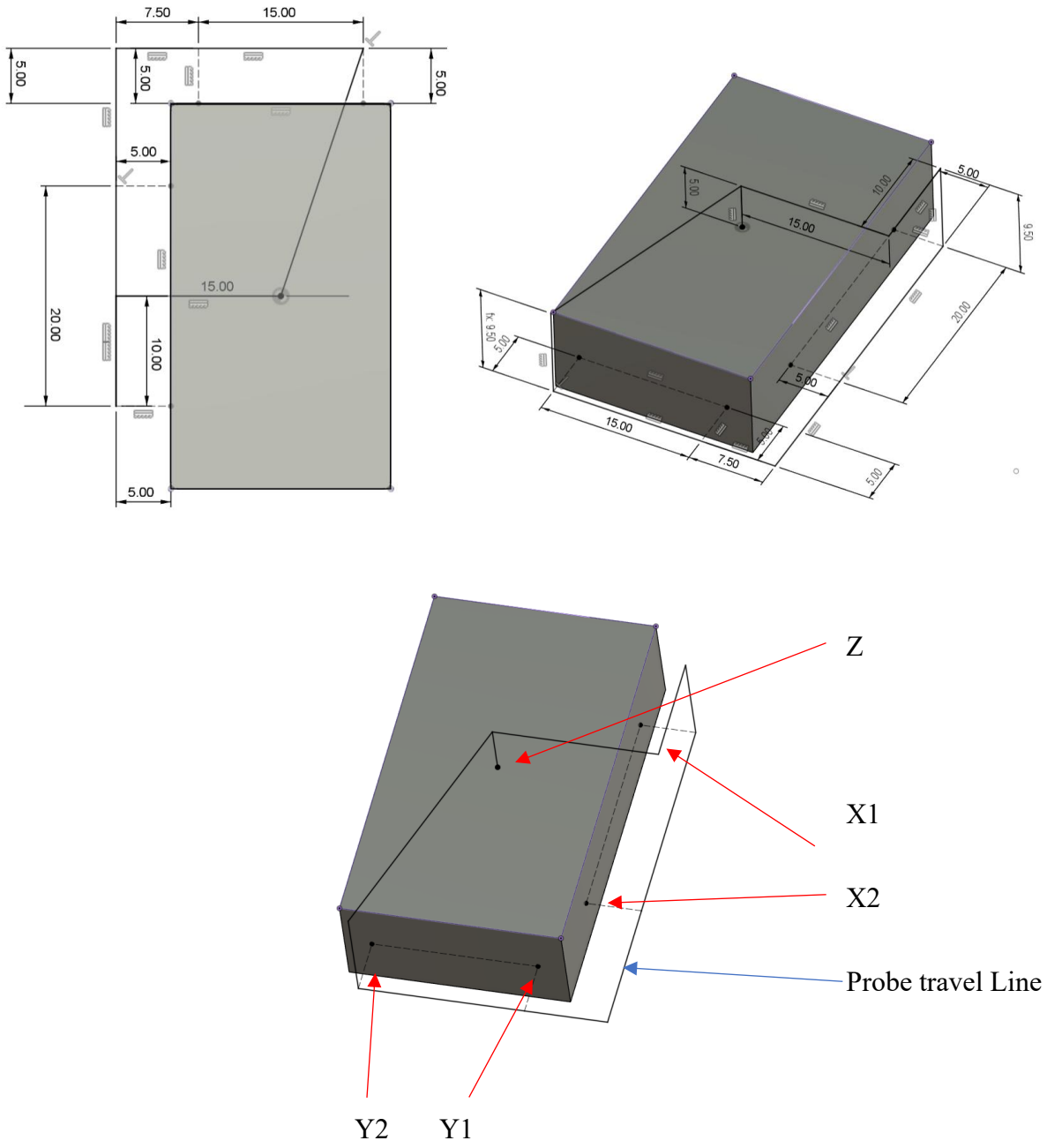


Figure 7-18 gauge block prob locations and probing tool path

The probe is controlled using the Aerotech A3200 system, which employs its proprietary programming language, AeroBasic. AeroBasic supports G-code and M-code commands, as well as programming logic, enabling precise and flexible control of the CNC system. The Hexagon M&H infrared probe (IRP40.50) operates wirelessly, transmitting a signal to a receiver connected to the CNC controller via digital inputs to indicate its triggered state. During

measurement, the ruby tip of the probe is moved into contact with the workpiece. Upon contact, the probe tip deflects, triggering a signal that instructs the CNC controller to stop and record the current location. The recorded position accounts for the probe radius, thereby allowing accurate determination of the edge contact point. To minimise probe location error, the feed rate during the probing approach is set to 10 mm/min. This low feed rate reduces the influence of dynamic motion errors and minimises probing forces that could otherwise contribute to measurement error. The AeroBasic code used to detect probe contact and control the measurement process is shown in Figure 7-19.

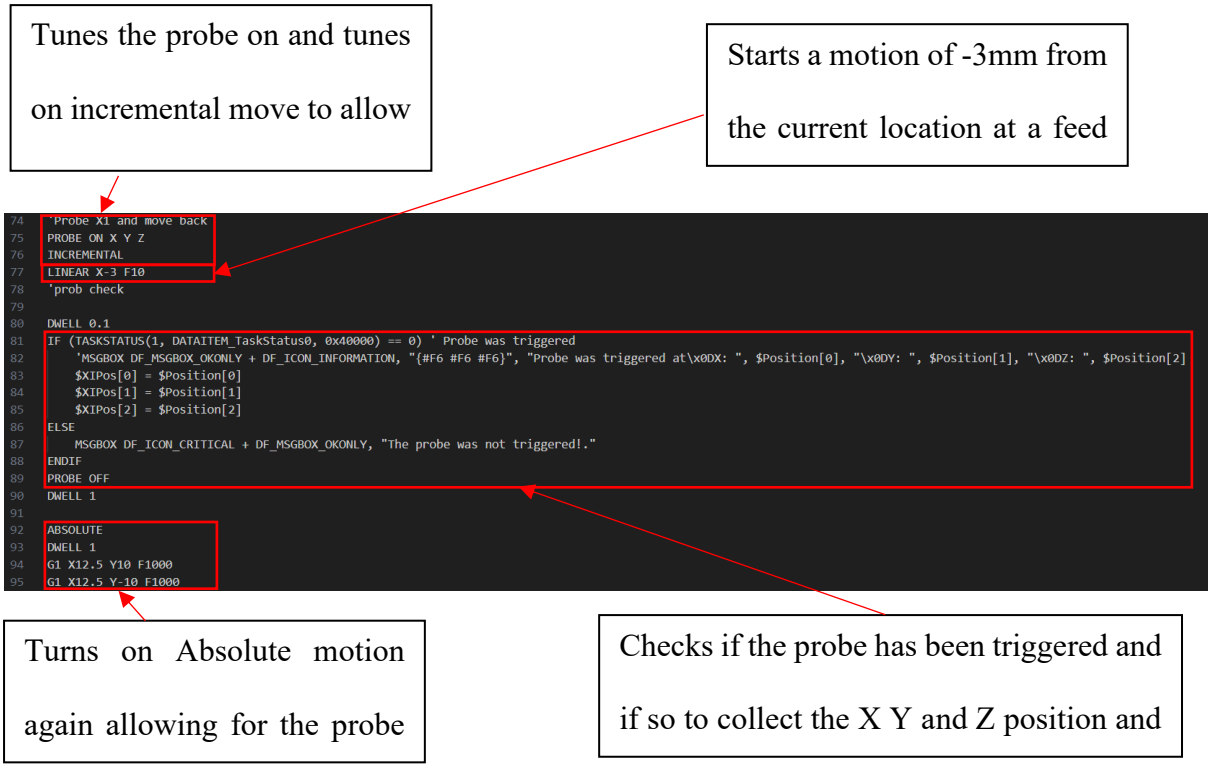


Figure 7-19 Aerobasic probing for repeated probing

This process is repeated for all predefined probe locations, ensuring that all necessary data points are collected. Upon completion of the program, the recorded values are displayed, representing the measured positions of the placed gauge block. These measurements are subsequently used to calculate the centre of the gauge block. The structure of the collected data and its presentation by the Aerotech system are illustrated in Figure 7-20.

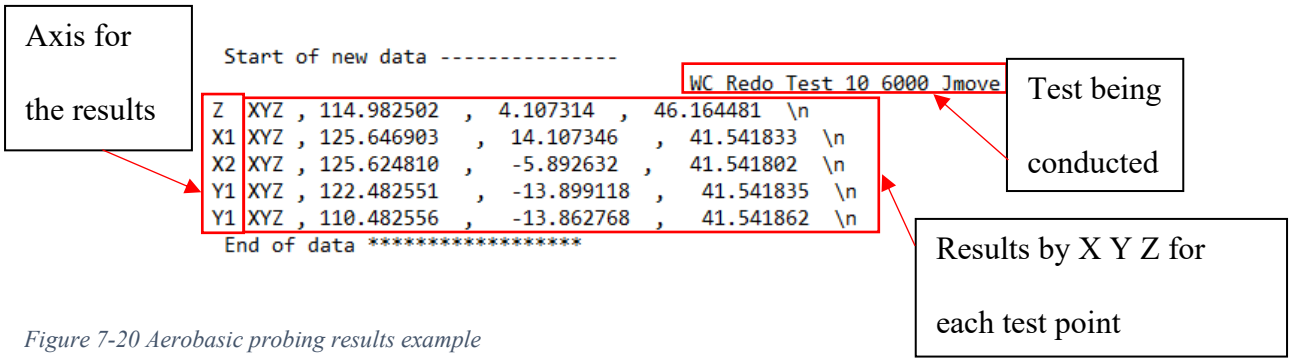


Figure 7-20 Aerobasic probing results example

Using this method to collect the data points, all the necessary information for each location can be captured. These measurements are then processed to calculate the centre location of the gauge block. The procedure for this calculation will be explained in the following section.

7.3.4 Data processing

As described earlier and illustrated in Figure 7-9, the method for identifying the centre of the gauge block involves measuring two points on each of the X and Y sides of the block. These points allow for the determination of the orientation of the block's sides. By knowing the distances between these points, the geometric centre of the block can be calculated.

Orientation errors are determined using trigonometry, as the lengths of the adjacent and opposite sides are known from the measured points. This calculation is illustrated in Figure 7-21 and defined mathematically in equation 7-1 for X-axis orientation and equation 7-2 for Y-axis orientation. As stated, the orientation error previously is only used to determine the centre of the gauge block and will not be an error to be tracked, as the proposed approach does not have the capability yet to correct for such errors.

$$(X\ Orintation\ Error)\theta = tan^{-1}\frac{(X2-X1)}{(Y2-Y1)} \quad (7\text{-}1)$$

$$(Y\ Orintation\ Error)\theta = tan^{-1}\frac{(Y1-Y2)}{(X1-X2)} \quad (7\text{-}2)$$

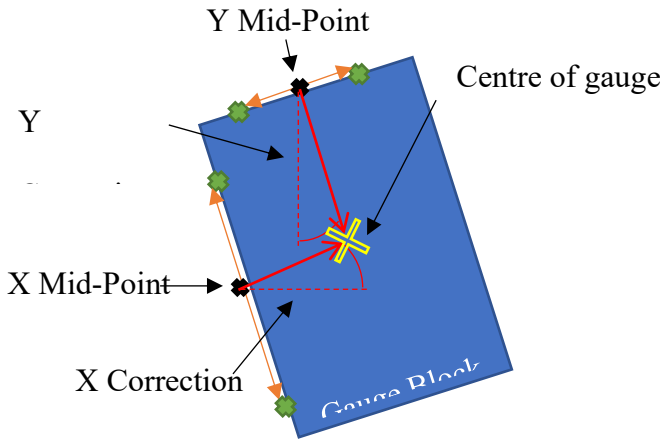


Figure 7-21 Gauge block centre location circulation illustration

Using the two measured points, the midpoint of the gauge block side can be determined, which allows for the calculation of the centre of that plane. By combining the known dimensions of the gauge block with the measured orientation angle, the overall centre of the block can be calculated. The XY coordinates of the centre are obtained by projecting from the midpoint using the radius defined as half the width of the gauge for the relevant side and the orientation angle, as illustrated in Figure 7-21 and expressed in equation 7-3 and equation 7-4 for the X and Y Plan Midpoint with equation 7-5 and equation 7-6 for the X and Y centre point.

$$X\ Plan\ Mid\ \text{Point} = \frac{(X1-X2)}{2} \quad (7\text{-}3)$$

$$Y\ Plan\ Mid\ Point = \frac{(Y1-Y2)}{2} \quad (7\text{-}4)$$

$$X\ Centre\ Point = X\ Plan\ Mid\ \text{Point} + \left(\frac{gauge\ X\ diemention}{2}\right) * Sin(X\ Rotation\ Error) \quad (7\text{-}5)$$

$$Y\ Centre\ Point = Y\ Plan\ Mid\ \text{Point} + \left(\frac{Gauge\ Y\ Diemention}{2}\right) * Sin(Y\ Rotation\ Error)$$

(7-6)

These calculations will be performed using Microsoft Excel to facilitate data organisation and result generation. This method of calculating the centre of the block leaves some uncertainty as to the accuracy of the probe location due to possible errors in the probing of the two, deviating from the centre line of the gauge block. If the placement error is too great, the calculated plane of the axis for the centre could be wrong. This approach was selected as it allows for fast testing, as ten repetitions will be conducted to validate the performance of the XAI predictions. A more accurate approach would be to probe an additional four points, mirroring the four selected, as the midpoint between the two opposite readings could determine the centre axis plan and from there the centre of the gauge block. This method was not used as the increase in probing time would double, and the uncertainty of the proposed method is not believed to be high.

Thermal compensation will be applied to account for the temperature of the gauge block at the time of probing. During repeated testing, the block temperature was observed to rise to 34.5 °C in a room that was measured to be 18.8 °C, primarily due to heating from the electromagnet. Using the manufacturer-specified thermal expansion coefficient of 5 µm/°C, the XYZ probing locations are adjusted accordingly, thereby compensating for the expansion of the steel and removing thermal-induced errors from the measurements.

7.3.5 Test results

The results of the gauge block placement experiment provide an in-depth assessment of the accuracy and precision of the proposed digital twin-driven dynamic error compensation approach for both movement types. To account for systematic setup errors in the COBOT and hybrid mill, an offset will be applied to the MoveJ and MoveL results. This offset is calculated

as the average of all collected results for the respective axis in each test, with the intention of eliminating setup-related errors from the jigs. This approach is considered appropriate, as these setup errors remain consistent throughout each batch of tests, given that the jigs are not moved or altered during testing.

7.3.5.1 Placement results

As previously discussed, the placement data have been processed to determine the centre locations of the gauge blocks. The results of the experiments for the MoveL and MoveJ motion types will be presented separately to enable a more detailed and comprehensive analysis of each motion type's performance.

7.3.5.1.1 MoveL Results

Using the collected data, the offset applied to all the data is shown in Table 7-5

Table 7-5 Move L applied average offset

X Offset (mm)	Y Offset (mm)	Z Offset (mm)
0.7273279	-0.53391271	0.04607305

The results of the probed locations with the measured offset have been applied and are shown in Figure 7-22.

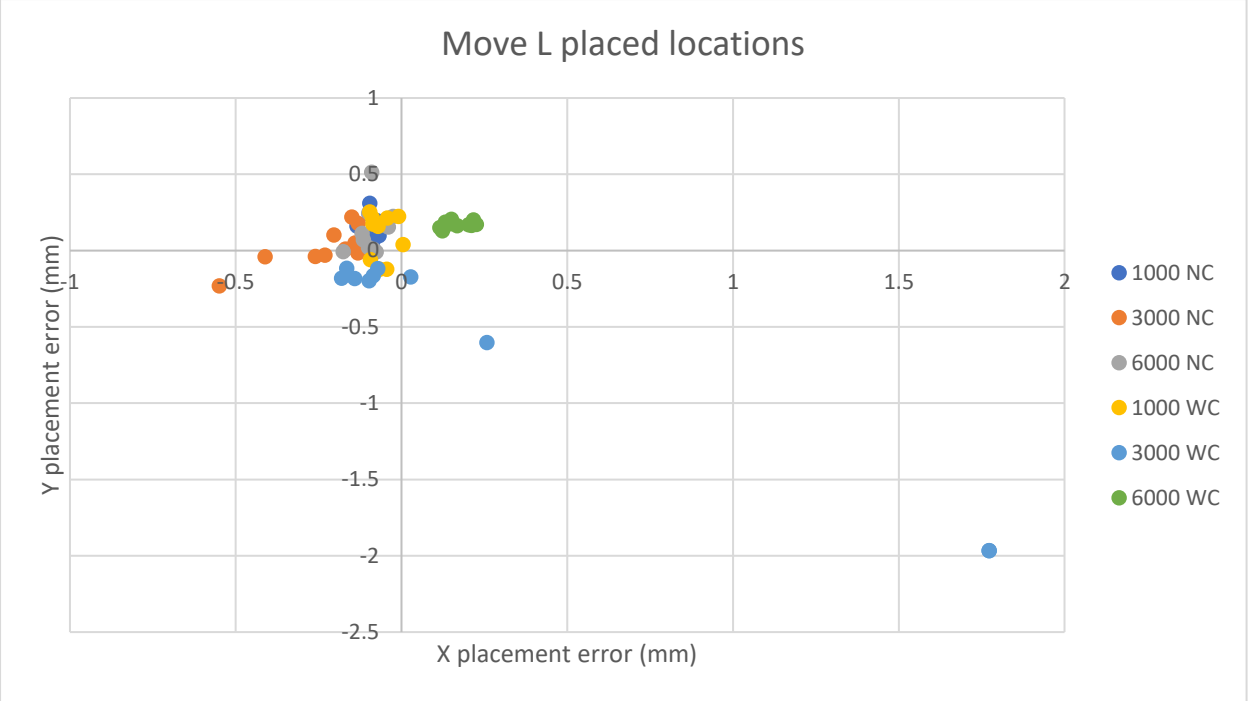


Figure 7-22 MOVE L place locations

It is evident from the measured data that some anomalies are present; these will be investigated further later in this section. The average calculated centre location of the gauge block for each feed rate, as well as the overall results for both NC and WC, is presented in Figure 7-23 and 7-23. The corresponding X and Y locations for each result are summarised in Table 7-6.

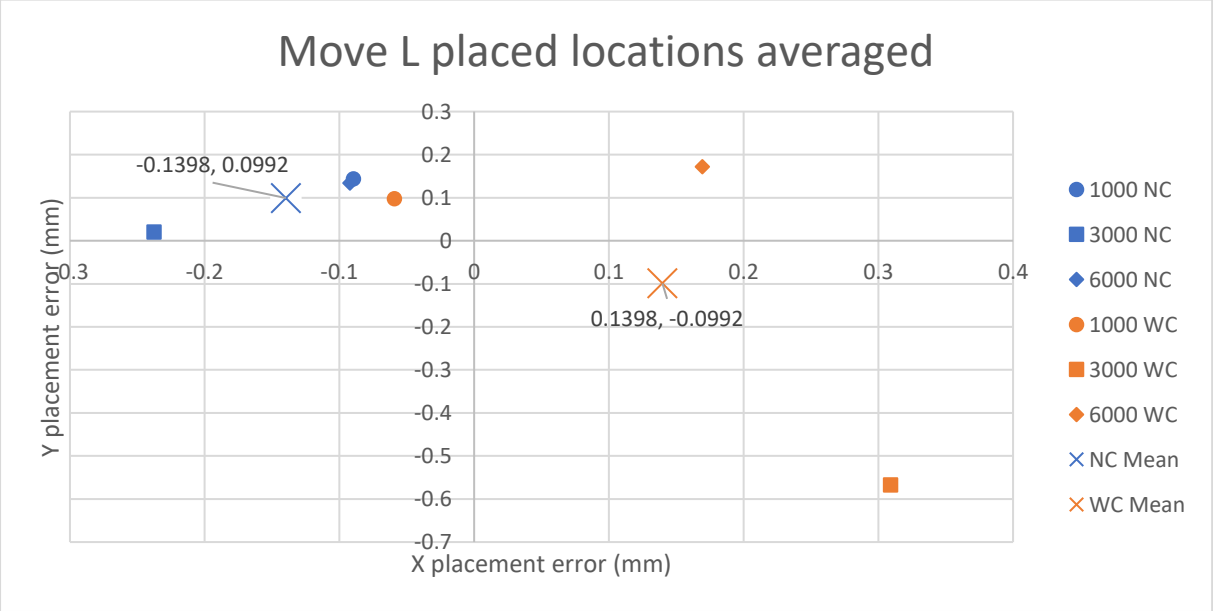


Figure 7-23 Move L placed locations averaged

Table 7-6 Move L X and Y placement error

Test Condition	X (mm)	Y (mm)
1000 NC	-0.089	0.144
1000 WC	-0.059	0.098
3000 NC	-0.238	0.020
3000 WC	0.309	-0.567
6000 NC	-0.092	0.134
6000 WC	0.169	0.172
All NC	-0.140	0.099
All WC	0.140	-0.099

To facilitate comparison of the results, the radial distance from the measured gauge block location to the designated centre (X_0, Y_0) was calculated. This provides a clear reference for assessing how accurately the gauge block was placed relative to the intended centre. The radial distance, C, was determined using the Pythagorean theorem.

This approach allows for a single metric to evaluate placement accuracy for all test points, with the radial errors being shown in Figure 7-24 and Table 7-7.

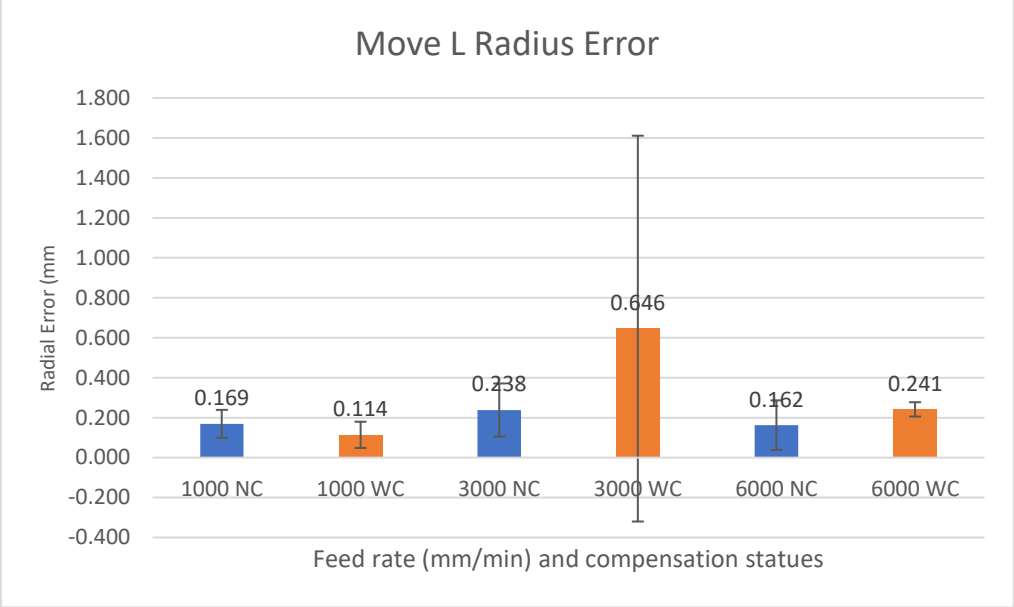


Figure 7-24 MoveL Radial Error

Table 7-7 Move L Radial Error

Test Conducted	Radial Error (mm)		
1000 NC	0.169		
3000 NC	0.238		
6000 NC	0.162	**Difference (mm)**	**Error Reduction (%)**
1000 WC	0.114	0.055	32.565
3000 WC	0.646	-0.407	-170.894
6000 WC	0.241	-0.079	-48.565
All NC	0.171		
All WC	0.171	8.88178E-16	0.0000

As observed, the corrective method did not perform as intended. A significant placement error was recorded in the 3000 mm/min test, and an increase in error was also evident at 6000

mm/min, with only the 1000 mm/min test showing a marginal improvement. As shown in Figure 7-22, Several measurements appear to be clear anomalies compared to the rest of the data. Consequently, further investigation was undertaken to identify potential causes and, where appropriate, remove outliers.

Analysis of the standard deviation provides additional insight. As summarised in Table 7-8The data does not indicate meaningful improvements from the digital twin relative to the system operating without compensation. In fact, in most cases, the standard deviation increased with the use of the digital twin, suggesting a reduction in consistency of placement across repetitions for each axis.

Table 7-8 Move L standard deviation

	SD (X)	Difference	SD (Y)	Difference	SD (Z)	Difference	Mean SD	Difference
1000 NC	0.018		0.077		0.015		0.028	
1000 WC	0.035	-0.017	0.136	-0.059	0.018	-0.003	0.052	-0.024
3000 NC	0.126		0.128		0.038		0.042	
3000 WC	0.742	-0.616	0.712	-0.585	0.026	0.012	0.330	-0.289
6000 NC	0.040		0.155		0.033		0.056	
6000 WC	0.039	0.001	0.021	0.134	0.037	-0.004	0.008	0.048

To begin investigating potential anomalies, the complete dataset collected during testing was plotted to facilitate easy identification of values that deviate from the general trend. The plotted data are presented in the MoveL dataset shown in Figure 7 25 MoveL Bar Chart Complete Data

Figure 7-25 MoveL Bar Chart Complete Data

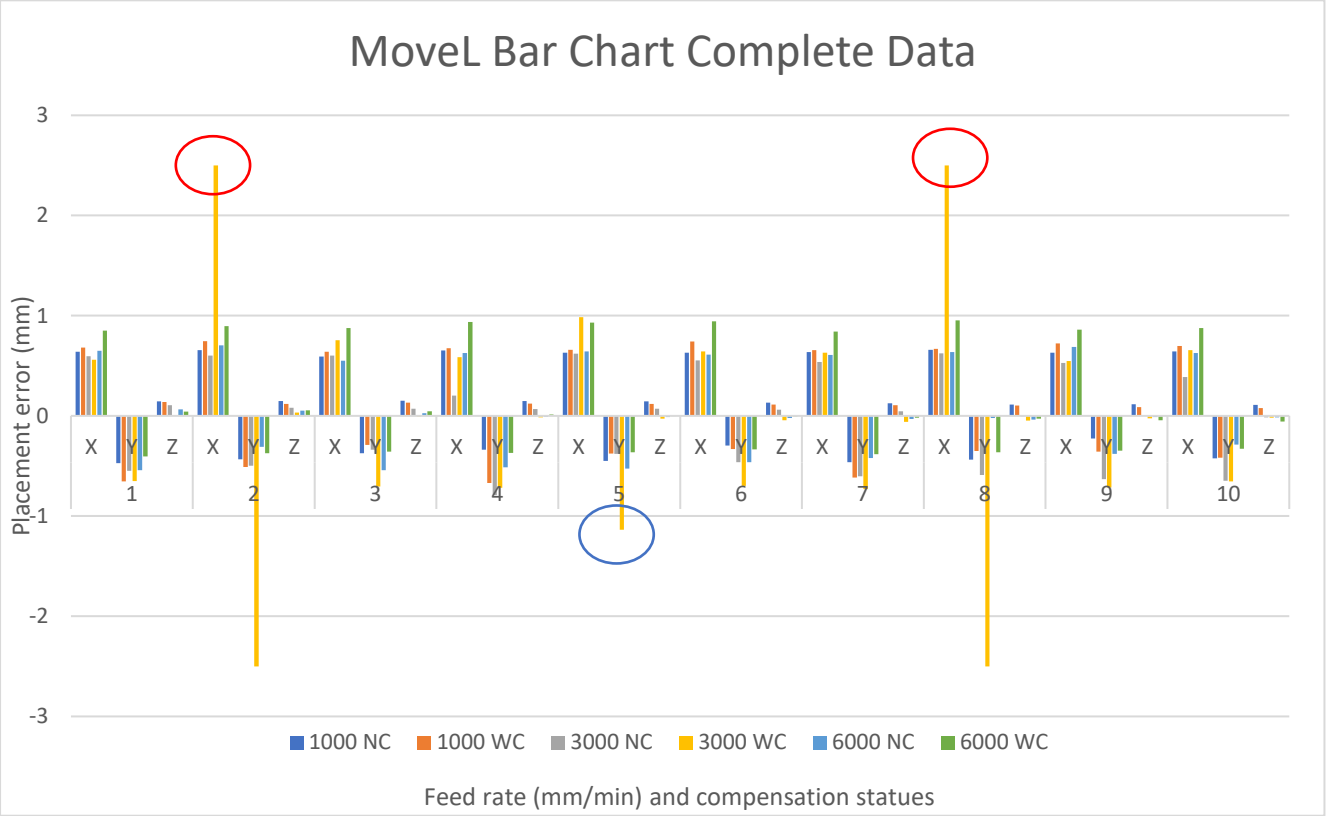


Figure 7-25 MoveL Bar Chart Complete Data

Three points have been identified as anomalies. Red circles indicate prediction errors, while blue circles represent setup errors. Prediction errors arise from inaccuracies in the XAI-predicted values, whereas setup errors are caused by inconsistencies in restarting the test with the gauge block positioned differently, which affects placement and worsens the results.

These two types of errors can be distinguished by examining the symmetry of the errors between the X and Y axes. Since the corrective action applies calculated corrections radially, both axes are adjusted equally. Therefore, if the errors are similar on both X and Y axes, they are likely due to prediction errors. Conversely, discrepancies between the axes suggest setup errors, where the gauge block was not placed precisely in the same location.

Based on this analysis, the three identified anomalies, two of which are believed to be prediction errors, are shown in Table 7-9. Test 5 is believed to be a setup error, as the discrepancy of 150 μm between the X and Y errors.

Table 7-9 Move L anomaly data points

3000 mm/min	X (mm)	Y (mm)	Difference (mm)
Test 2	2.5003	-2.5000	-0.0002
Test 5	0.9854	-1.1367	0.1513
Test 8	2.5001	-2.4998	-0.0003

With the anomalies identified, the data were reprocessed with these points removed. Notably, these occurred at the same feed rate of 3000mm/min, corresponding to previous issues observed during compensated testing in the XYZ validation experiments. This suggests a potential pattern related to the XAI prediction model's performance at this feed rate, which will be examined further later in this section.

After removing the anomalies, the resulting data points are presented in Figure 7-26.

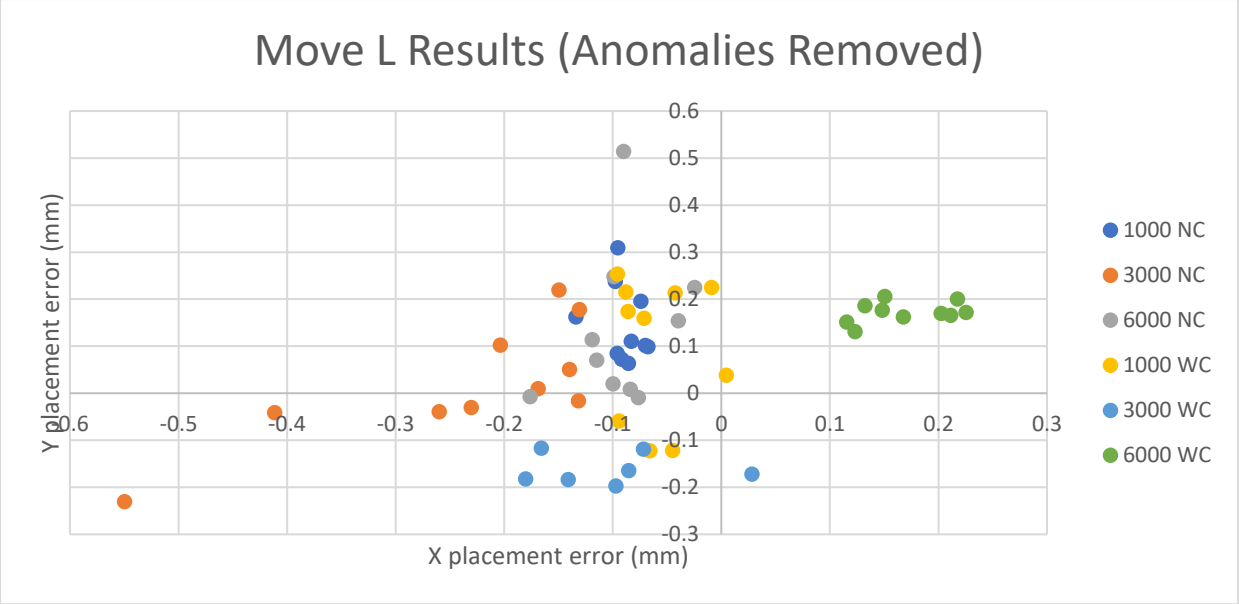


Figure 7-26 Move L Results (anomalies removed)

It can be seen that the processed data are much clearer, although a few values in the 3000 mm/min NC dataset still appear slightly offset. However, since these values do not deviate significantly from the overall raw data, they are assumed not to be anomalies.

Using this revised dataset, the average placement locations were calculated and are presented in 27, with the corresponding numerical values provided in Table 7-10.

Figure 7-27 Move L averaged results (anomalies removed)

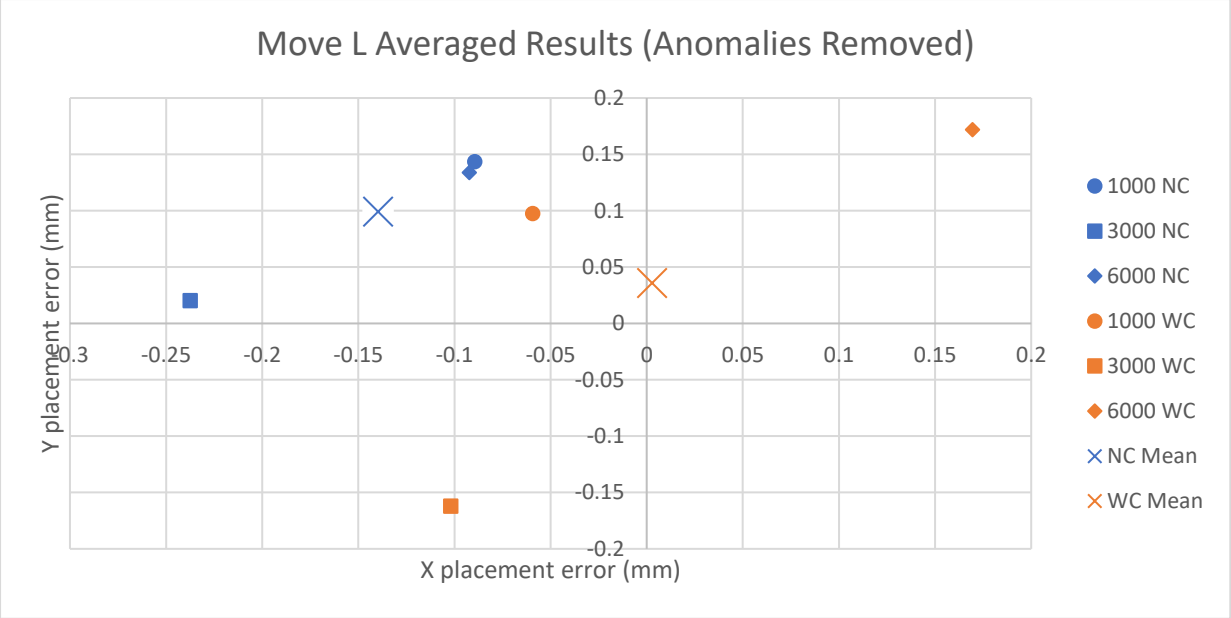


Figure 7-27 Move L averaged results (anomalies removed)

Table 7-10 Move L X and Y placement errors (anomalies removed)

Test Condition	X (mm)	Y (mm)
1000 NC	-0.089	0.144
1000 WC	-0.059	0.098
3000 NC	-0.238	0.020
3000 WC	0.309	-0.567
6000 NC	-0.092	0.134
6000 WC	0.169	0.172
All NC	-0.140	0.099
All WC	0.003	0.036

Using the revised dataset, the radial error for each test was calculated to facilitate a direct comparison between the NC and WC cases. The results are presented in Figure 7-28 MoveL radial results (anomalies removed)

28, with the corresponding numerical values, differences, and percentage error reductions summarised in Table 7-11.

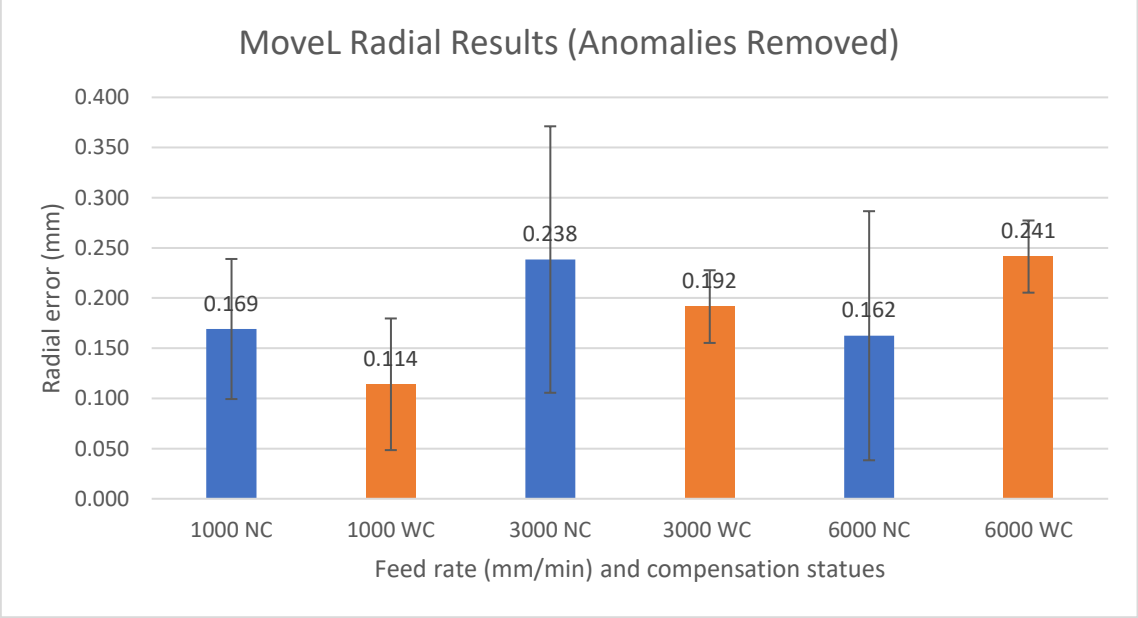


Figure 7-28 MoveL radial results (anomalies removed)

Table 7-11 Move L Radial Error (anomalies removed)

	Radial error of MoveL data (mm)		
1000 NC	0.169		
3000 NC	0.238		
6000 NC	0.162	**Difference (mm)**	**Error Reduction (%)**
1000 WC	0.114	0.055	32.565
3000 WC	0.192	0.047	19.656
6000 WC	0.241	-0.079	48.565
All NC	0.171		
All WC	0.036	0.136	79.082

The proposed approach improved the COBOT's placement accuracy at feed rates of 1000 mm/min and 3000 mm/min, achieving error reductions of 19.7% and 32.6%, respectively. However, at 6000 mm/min, an increase in error of 48.6% was observed, indicating a performance degradation that warrants further investigation.

Overall, when the position data for X and Y are averaged and the radial error calculated, the digital twin demonstrates a positive impact on the COBOT's accuracy across a range of feed rates in a pick-and-place task. This is notable, considering that the XAI system was not trained

using data from MoveL commands, highlighting the system's ability to generalise beyond its training conditions.

The standard deviation for the MoveL tests, after removing the identified anomalies, has been recalculated, with updated values presented in Table 7-12. While the recalculated deviations show some reduction, they remain higher than those of the COBOT system without applying the proposed dynamic error compensation approach in most cases, suggesting that consistency has not improved substantially despite the reduction in mean error.

Table 7-12 Move L standard deviation (anomalies removed)

	SD (X)	Difference	SD (Y)	Difference	SD (Z)	Difference	Mean SD	Difference
1000 NC	0.018		0.077		0.015		0.028	
1000 WC	0.035	-0.017	0.136	-0.059	0.018	-0.003	0.052	-0.024
3000 NC	0.126		0.128		0.038		0.042	
3000 WC	0.292	-0.166	0.320	-0.193	0.026	0.012	0.132	-0.091
6000 NC	0.040		0.155		0.033		0.056	
6000 WC	0.039	0.001	0.021	0.134	0.037	-0.004	0.008	0.048

7.3.5.2 Move J

The offset for the MoveJ test has been calculated and shown in Table 7-13.

Table 7-13 Move J applied offsets

X Offset	Y Offset	Z Offset

0.623	-0.560	-1.064E-17

The results of the probed locations with the measured offset applied are shown in Figure 7-29.

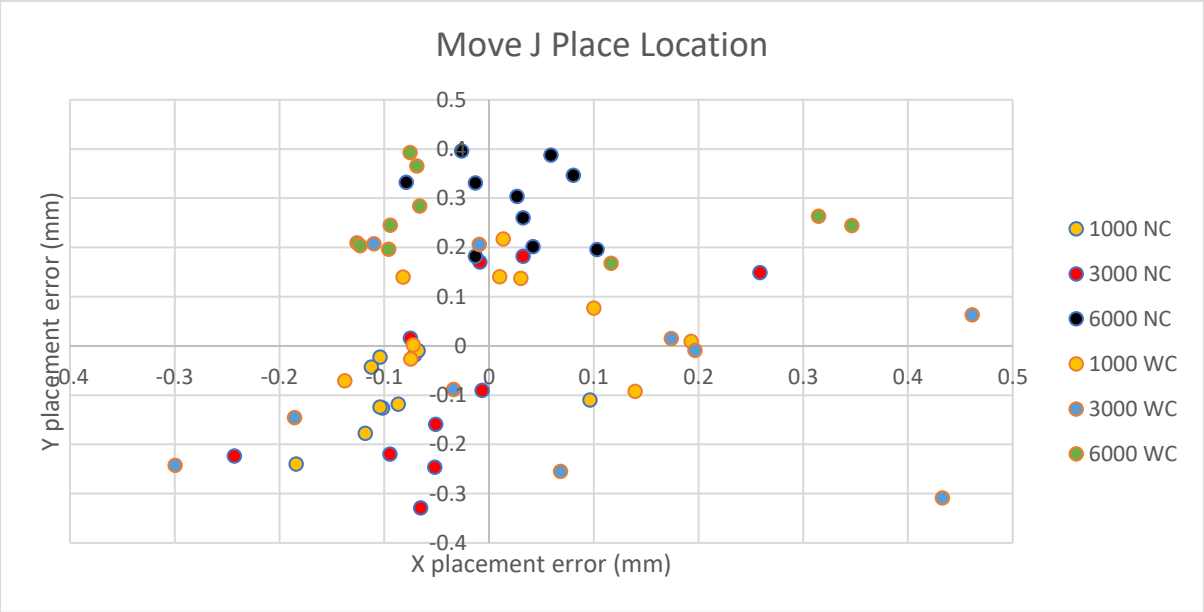


Figure 7-29 Move J place location

The averaged results for each feed rate can be seen in Figure 7-29Figure 7-30 with the values for each test shown in Table 7-14.

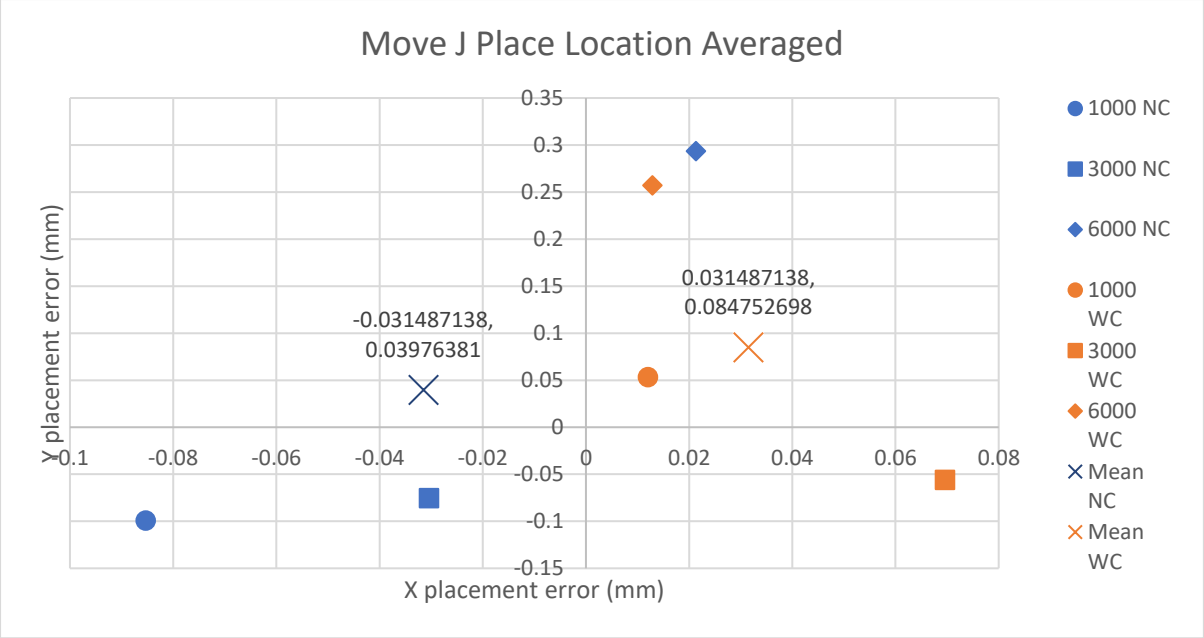


Figure 7-30 Move J place location averaged

Table 7-14 Move J X and Y errors

	X (mm)	Y (mm)
1000 NC	-0.085	-0.099
3000 NC	-0.030	-0.075
6000 NC	0.021	0.294
1000 WC	0.012	0.053
3000 WC	0.070	-0.056
6000 WC	0.013	0.257
All NC	-0.031	0.040
All WC	0.031	0.085

To enable a direct comparison of all results, the radial error for each test average was calculated using the same methodology as for the MoveL experiments. This approach provides a clear and consistent metric for assessing placement accuracy. The resulting radial errors, along with the corresponding differences and error reductions, are presented in Figure 7-31 and Table 7-15.

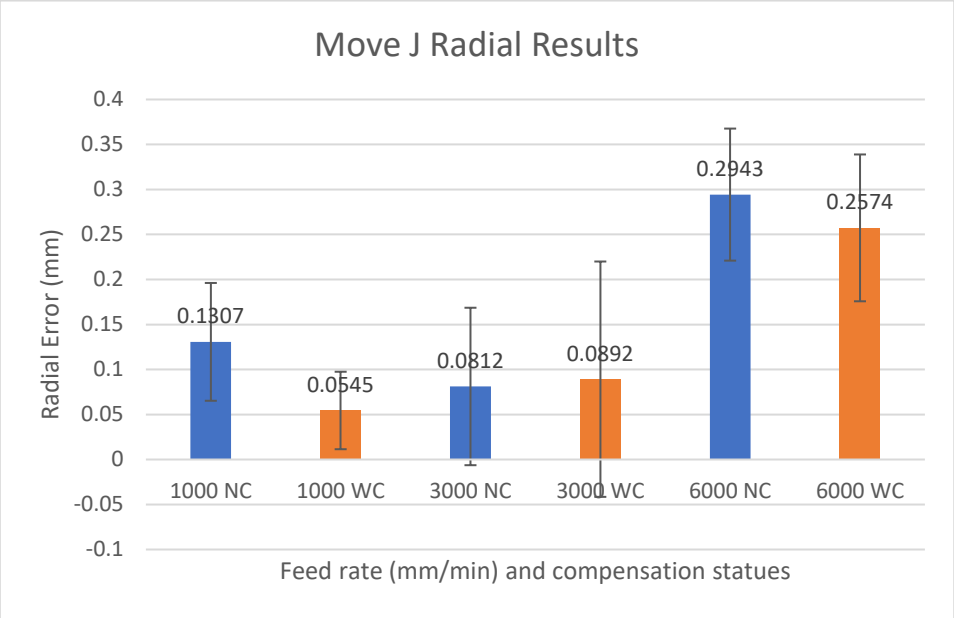


Figure 7-31 Move J radial error results

Table 7-15 Move J Radial Error

Test	Radial results (mm)		
1000 NC	0.131		
3000 NC	0.081	**Difference (mm)**	**Error Reduction (%)**
6000 NC	0.294		
1000 WC	0.054	0.076	58.313
3000 WC	0.089	-0.008	-9.971
6000 WC	0.257	0.037	12.560

As shown in Figure 7-31, the average radial results by feed rate indicate positive improvements at 1000 mm/min and 6000 mm/min, with only the 3000 mm/min test showing a deterioration in performance. Specifically, the 1000 mm/min and 6000 mm/min feed rates demonstrated error reductions of 58% and 12.5%, respectively, while the 3000 mm/min test exhibited an error increase of approximately 10%.

The standard deviation of these results has been calculated to assess consistency and is presented in Table 7-16.

Table 7-16 Move J Standard Deviation

XYZ

	SD (X)	Difference	SD (Y)	Difference	SD (Z)	Difference	Mean SD	Difference
1000 NC	0.068		0.072		0.022		0.023	
1000 WC	0.101	-0.034	0.099	-0.027	0.028	-0.006	0.034	-0.011
3000 NC	0.119		0.181		0.022		0.066	
3000 WC	0.237	-0.118	0.175	0.006	0.038	-0.017	0.083	-0.017
6000 NC	0.052		0.075		0.031		0.018	
6000 WC	0.172	-0.120	0.069	0.006	0.032	-0.001	0.059	-0.041

It can be seen that the standard deviation improved in only two instances across all tests, indicating that the digital twin did not consistently enhance the COBOT's placement accuracy.

This outcome is expected, as the deviation from the intended location varies randomly depending on the measured accelerations and the predictions generated by the trained equation.

It is believed that these variations are caused by anomalies in the data, similar to those observed in the MoveL tests, and will be investigated using a comparable approach. To begin this analysis, all measured probe points have been plotted, as illustrated in Figure 7-32. This visualisation allows for easy identification of values that deviate significantly from the rest, with red circles highlighting likely prediction errors and blue circles indicating potential setup errors.

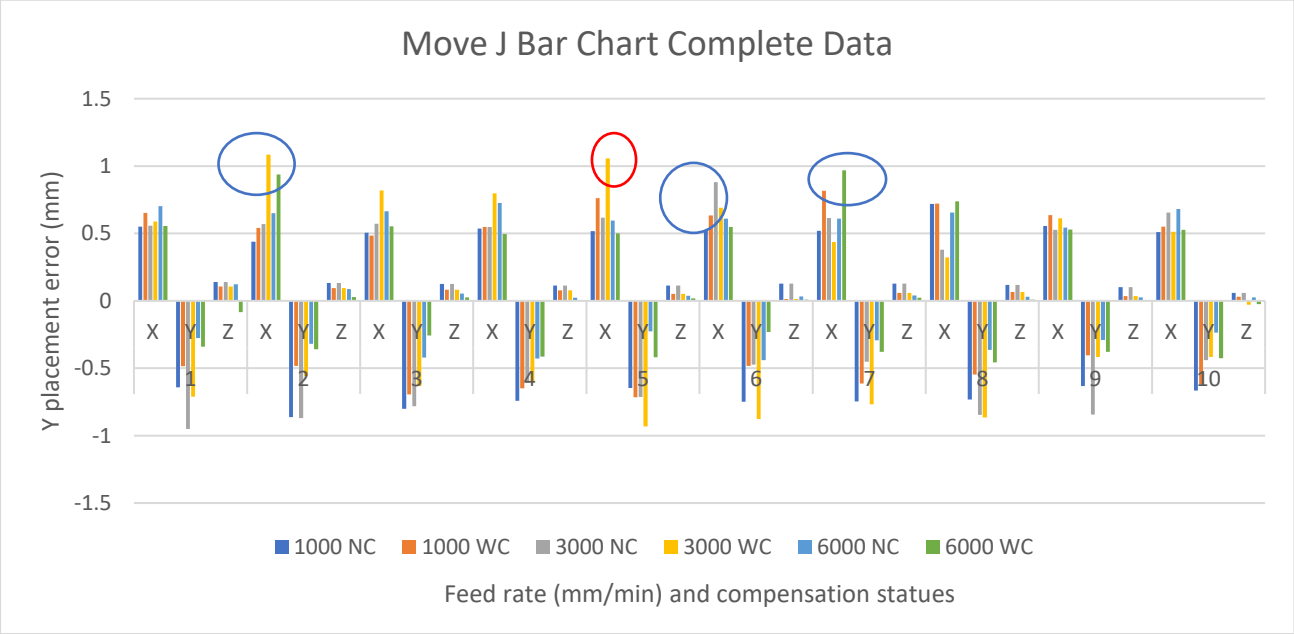


Figure 7-32 Move J bar chart of complete data

As previously discussed, prediction errors can be identified by their symmetrical nature in the X and Y axes, since the COBOT predictive digital twin applies the same corrective value for both axes when predicting radial error. In contrast, setup errors arise from inconsistencies in the initial placement of the gauge block, where it is not returned to the same position at the start of each test.

For this experiment, the majority of errors are believed to result from setup inconsistencies, as the X and Y deviations do not correspond. Specifically, these occur in 3000 WC Test 2, 3000 NC Test 6, 1000 WC Test 7, and 6000 WC Test 7. The only test showing indications of a prediction error is 3000 WC Test 5, where the X and Y deviations are similar.

The identified anomalies are summarised in Table 7-17, with potential prediction errors highlighted in yellow and setup errors in blue. Most errors are associated with setup inconsistencies. To ensure accurate evaluation of the digital twin, these anomalous values will be removed from the dataset. Regardless of the source of these errors, they do not reflect the system's intended performance. The underlying causes of these anomalies, particularly the recurring pattern observed at 3000 mm/min across validation, MoveL, and MoveJ tests, will be investigated in a later section.

Table 7-17 Move J anomalies data points

Test Number	**2**			**5**			**6**			**7**		
Axis error (mm)	**X**	**Y**	**Z**	**X**	**Y**	**Z**	**X**	**Y**	**Z**	**X**	**Y**	**Z**
1000 WC										0.816	-0.613	0.059
3000 NC							0.881	-0.474	0.128			
3000 WC	1.084	-0.559	0.094	1.055	-0.932	0.051						
6000 WC	0.937	-0.360	0.028							0.969	-0.378	0.024

As shown in Table 7-17, the values from the tests are similar to each other but not always symmetrical, supporting the conclusion that the majority of these errors arise from setup inconsistencies. This suggests that a new jig, capable of more repeatable and precise placement, would be required for further testing to evaluate the system under more controlled conditions.

The identified anomalous data points have been removed to assess the performance of the proposed dynamic error compensation approach under ideal conditions. The resulting dataset, after removal of these anomalies, is presented in Figure 7-33 Move J results (anomalies removed)

33.

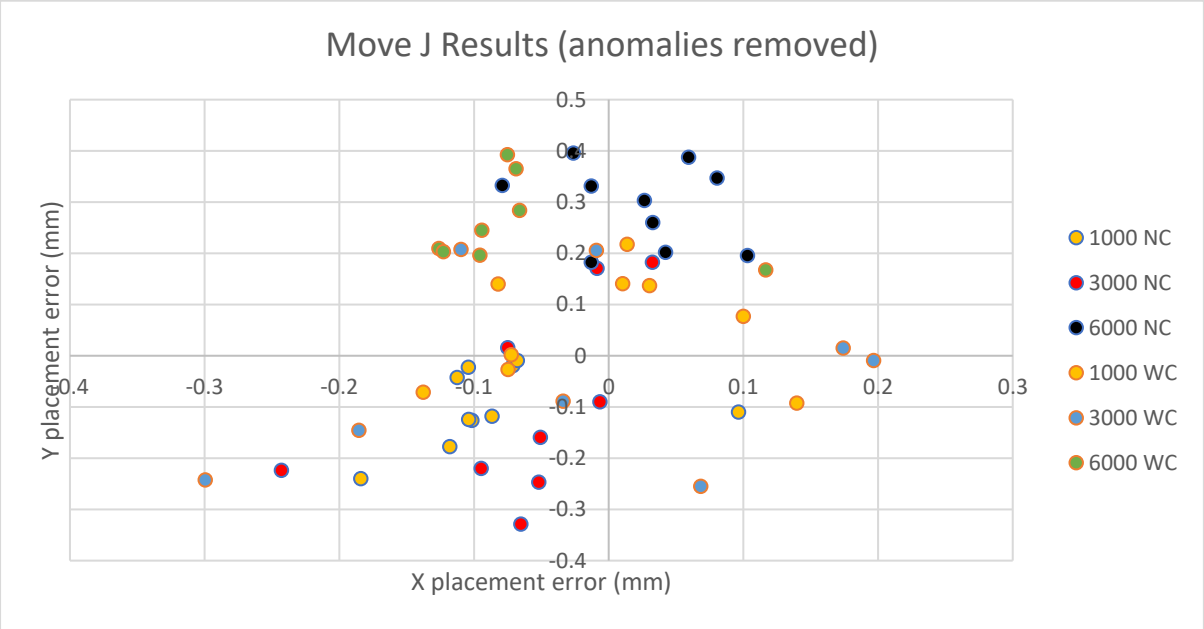


Figure 7-33 Move J results (anomalies removed)

With these anomalous values removed, the averaging of the complete dataset yields predominantly positive results. This is illustrated in Figure 7-34, with the corresponding average locations summarised in Table 7-18.

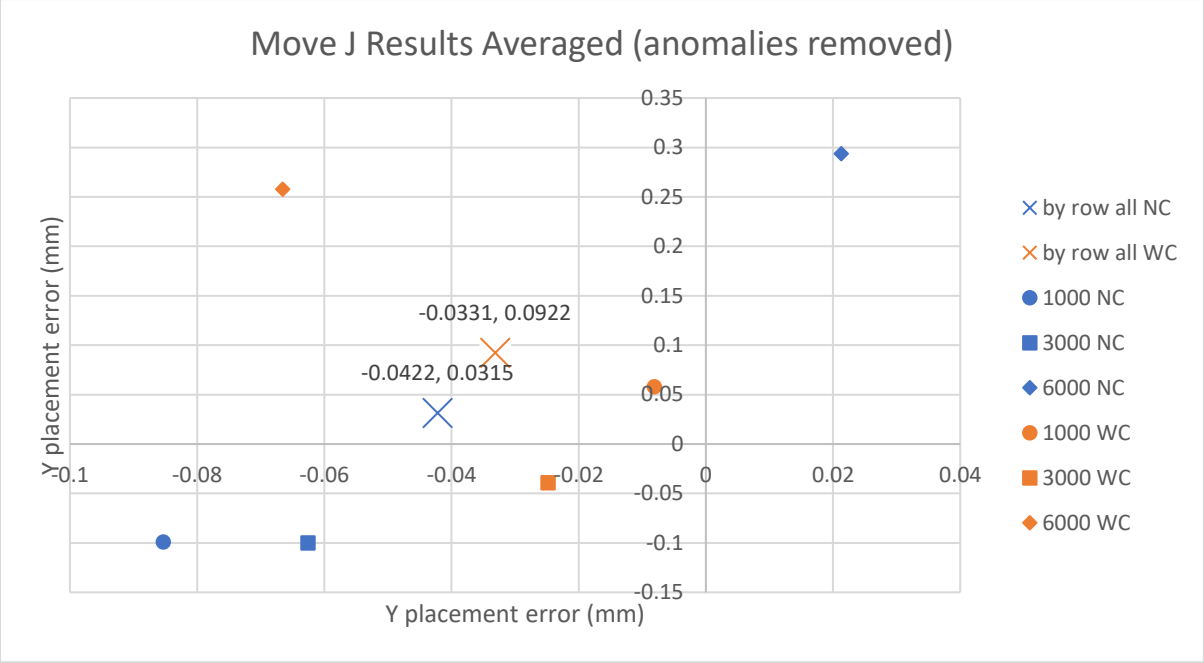


Figure 7-34 Move J Results averaged (anomalies removed)

Table 7-18 Move J X and Y error (anomalies removed)

Test	X (mm)	Y (mm)
1000 NC	-0.085	-0.099
3000 NC	-0.063	-0.100
6000 NC	0.021	0.294
1000 WC	-0.008	0.058
3000 WC	-0.025	-0.039
6000 WC	-0.067	0.258
All NC	-0.042	0.031
All WC	-0.033	0.092

Using the newly identified average locations, the radial error for each test was calculated to enable direct comparison between results. The radial errors are presented in Figure 7-35, with the corresponding values, differences, and error reductions summarised in Table 7-19.

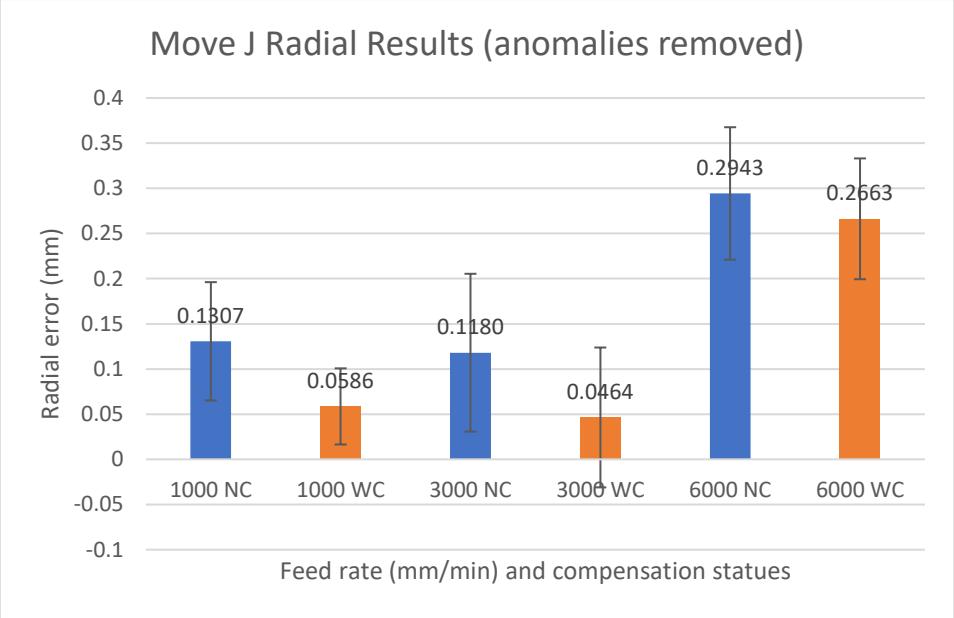


Figure 7-35 Move J Radial Results (anomalies removed)

Table 7-19 Move J Radial Error (anomalies removed)

Test	Anomalies removed (mm)	Difference (mm)	Error Reduction (%)
1000 NC	0.131		
3000 NC	0.118		
6000 NC	0.294		
1000 WC	0.059	0.072	55.179
3000 WC	0.046	0.072	60.715
6000 WC	0.266	0.028	9.527
NC	0.053		
WC	0.098	-0.045	86.183

With the anomalous data removed, the results show an error reduction ranging from 9.5% to 60%, with positive improvements observed at all feed rates when the X and Y locations are

considered individually. This demonstrates that, on average, the digital twin improves the accuracy of the COBOT in pick-and-place tasks across a range of feed rates when using MoveJ.

However, when the X and Y locations are averaged, and the radial error is calculated, the improvement is not as clear, despite all individual results showing positive gains. The cause of this discrepancy requires further investigation.

The standard deviation of the COBOT with the digital twin, calculated after removing anomalies, is presented in Table 7-20.

Table 7-20 Move J Standard Deviation (anomalies removed)

	SD (X)	Difference	SD (Y)	Difference	SD (Z)	Difference	Mean SD	Difference
1000 NC	0.068		0.072		0.022		0.023	
1000 WC	0.202	-0.134	0.195	-0.124	0.034	-0.012	0.078	-0.055
3000 NC	0.182		0.273		0.040		0.096	
3000 WC	0.279	-0.097	0.304	-0.032	0.041	-0.001	0.118	-0.023
6000 NC	0.052		0.075		0.031		0.018	
6000 WC	0.232	-0.180	0.161	-0.086	0.029	0.001	0.084	-0.066

As shown in Table 7-20, the standard deviation has increased across all tests and axes, indicating that while the system may have some impact on precision, it does not improve the overall accuracy of the COBOT.

7.3.6 Results and discussion

From this testing, it is clear that the proposed digital twin-driven dynamic error compensation approach for the COBOT produces positive results for the majority of the tests conducted, with the summarised results presented in Table 7-21 for MoveL and Table 7-22 for MoveJ.

Table 7-21 COBOT Move L pick and place results

Move L Test	Radial Error with anomalies (mm)	Difference (mm)	Error Reduction (%)	Radial Error Anomalies removed (mm)	Difference (mm)	Error Reduction (%)
1000 NC	0.169			0.169		
3000 NC	0.238			0.238		
6000 NC	0.162			0.162		
1000 WC	0.114	0.055	32.565	0.114	0.055	32.565
3000 WC	0.646	-0.407	-170.894	0.192	0.047	19.656
6000 WC	0.241	-0.079	-48.565	0.241	-0.079	-48.565
Average NC	0.190			0.190		
Average WC	0.334	0.000	0.000	0.182	0.008	4.044

Table 7-22 COBOT Move J pick and place results

Move J Test	Radial Error with anomalies (mm)	Difference (mm)	Error Reduction (%)	Radial Error Anomalies removed (mm)	Difference (mm)	Error Reduction (%)
1000 NC	0.131			0.131		
3000 NC	0.081			0.118		
6000 NC	0.294			0.294		
1000 WC	0.054	0.076	58.313	0.059	0.072	55.179
3000 WC	0.089	-0.008	-9.971	0.046	0.072	60.715
6000 WC	0.257	0.037	12.560	0.266	0.028	9.527
Average NC	0.169			0.181		
Average WC	0.134	0.035	20.763	0.124	0.057	31.642

The results demonstrate that the proposed approach can improve the accuracy of the COBOT for both movement types, with an average error reduction across all testing (with anomalies removed) of 31.6% and 4% for MoveJ and MoveL, respectively. However, these results are influenced by the presence of anomalies, which were identified as either prediction errors or setup errors. Across all tests, six setup errors and three prediction errors were detected. Set-up errors are attributed to inconsistencies in the jig or human operation, whereas the prediction errors consistently occurred at the 3000 mm/min feed rate with compensation.

The repetition of prediction errors at 3000 mm/min across the MoveL, MoveJ, and validation experiments suggests a potential limitation in the XAI model at this specific speed. This may be due to insufficient training data at this feed rate. By contrast, at 1000 mm/min—which was also not part of the training data—the COBOT appears more stable, likely because the slower motion reduces the magnitude of dynamic errors. A potential solution is to expand the training dataset to include a wider range of COBOT feed rates, enabling the XAI system to develop a more comprehensive understanding of the COBOT's dynamics.

When comparing the accuracy of the COBOT with and without compensation, the results indicate that both movement types were able to reduce placement error, resulting in an overall improvement in accuracy. For MoveL, an improvement was observed at a feed rate of 1000 mm/min, while MoveJ demonstrated improved accuracy at 1000 mm/min and 6000 mm/min, including the anomalous data points. These results suggest that, for MoveJ, the use of an explainable artificial intelligence (XAI) model combined with accelerometer data provides a clear improvement in positional accuracy. However, further investigation is required to understand why the compensation method did not improve performance at 3000 mm/min for either movement type. The prediction errors discussed can be identified by the placement error appearing equally in both the X and Y axes, and the underlying cause of this behaviour must be investigated in order to improve the effectiveness of the proposed approach. One possible

explanation is the presence of unexpected noise during the motion tests. Improvements to data filtering methods or further refinement of the model training process may therefore lead to more accurate predictions.

Another contributing factor may be the nature of the training motion and the application of corrective action. The training data was collected using a ball bar, resulting in circular motion with ground truth based on radial error from the centre. The dynamics of the COBOT, however, vary depending on movement direction and the interactions of its six axes. Expanding the training data to include diverse motion types would allow for axis-specific corrective actions rather than a purely radial prediction. Currently, this limitation arises from the reliance on ball bar testing.

Addressing this issue could involve either extending the range of feed rates included in XAI training or creating a new dataset to enable axis-specific corrective actions.

Despite these limitations, the positive results indicate that the proposed approach is effective in achieving improvement in accuracy and thereby meeting the objectives set for its development. Additionally, the digital twin offers potential benefits for production efficiency through automated loading and could support lights-out manufacturing by enabling precise control and monitoring of COBOT motion. These potential applications will be explored further in the following chapter.

7.4 Summary

In this chapter, the effectiveness of the proposed digital twin-driven dynamic error compensation approach for three-dimensional corrective action in COBOT was evaluated. The validation was conducted in two stages to ensure that the system could be reliably assessed, as the XAI dynamic error prediction model had been trained using only two-dimensional data.

The first stage involved a validation test in which the COBOT moved a ball bar along three circular paths at varying heights, with transition movements between them. The programmed path maintained a fixed radius from the centre point, allowing for the measurement of radial error throughout the complete motion. The results demonstrated a reduction in radial error of 55.35% and a decrease in standard deviation of 59.39%, confirming the effectiveness of the proposed approach in compensating for dynamic errors in motion.

Following the successful validation, a pick-and-place experiment was conducted to evaluate the digital twin in a more realistic task for which it was designed. The experiment included both movement types: MoveL and MoveJ, to assess the limits of the system when operating with motions that had not been included in the training data. To facilitate consistent and repeatable placement, jigs were used to allow rapid resets, and a steel gauge block was selected as the object due to its high dimensional accuracy, which minimised potential errors associated with the object itself.

The position of the gauge block was measured using a Hexagon touch probe, recording two points along the X-axis and two along the Y-axis to account for orientation errors. These measurements were then used to calculate the centre of the gauge block and the radial error relative to the origin of the hybrid mill. The pick-and-place tests were conducted at feed rates of 1000, 3000, and 6000 mm/min, with and without compensation, and each test was repeated ten times to assess the reliability of the XAI system across multiple iterations.

During the analysis, several anomalies were identified across both WC and NC tests. Further investigation determined these anomalies were due to two primary causes:

Set-up errors – resulting from the gauge block not being returned to the exact start location. These errors were identifiable by large discrepancies between the X and Y measurements.

Prediction errors – arising from the XAI model, which applies radial corrections equally to both axes. These were identifiable by similarly large errors in X and Y measurements.

Out of 120 tests conducted, nine anomalies were detected, comprising three prediction errors and six setup errors. The six setup errors could occur in any test and represent human error, while prediction errors were only possible in the 60 tests using compensation, leading the three-prediction error to have a 5% likelihood of happening in the tests. Specifically, two prediction anomalies occurred in MoveL and one in MoveJ, likely reflecting the limitations of the training data, which were collected from circular motions and did not include linear movements.

Results, including all anomalies, were initially calculated to compare the system performance under standard operating conditions. Once anomalies were removed, the digital twin demonstrated improved placement accuracy in all tests except for MoveL at 6000 mm/min. On average, the proposed dynamic error compensation approach reduced the radial placement error by 4% for MoveL and 31.6% for MoveJ. The lower improvement in MoveL is expected, as the XAI was trained only on circular motions, and linear motion was not included in the training dataset.

For MoveJ, the proposed approach showed substantial improvements, with the highest error reduction of 60.7% observed at 3000 mm/min. This result is notable because this feed rate was not included in the training data and had the highest occurrence of prediction errors, suggesting that the removal of anomalies may have contributed to the observed improvement.

These findings indicate that further research is warranted to expand the training dataset. Currently, the training data is limited to circular motions at a narrow range of feed rates, resulting in the corrective action being applied radially rather than per axis. Collecting more comprehensive data that captures the COBOT's dynamics across different axes would enable

the application of axis-specific corrective actions, providing more precise and targeted compensation for dynamic errors.

8 Chapter 8– A Digital Twin-Driven Ultra-Precision Machining System

8.1 Introduction

The research in the previous chapter on a predictive digital twin-driven dynamic error compensation approach for the hybrid mill and COBOT lays a solid foundation to establish a smart manufacturing system capable of performing real-time corrections for dynamic errors during ultra-precision machining. The smart manufacturing system integrates available machine data with additional sensory information from accelerometers and applies machine learning-based predictive models to estimate and compensate for dynamic deviations. This proposed dynamic error compensation approach was supposed to enhance the accuracy of manufacturing operations by allowing corrective actions to be applied outside of the primary control loop using continuously updated process data.

Experimental results in the previous chapter demonstrated that the proposed digital twin-driven dynamic error compensation concept showed partial success, particularly in the case of the COBOT, where measurable improvements in placement accuracy were achieved. However, the implementation of the hybrid mill showed limited improvement, indicating that further refinement and development are required to achieve the desired level of precision and control.

This proposed demonstration of a fully digital twin–driven ultra-precision machining system, if implemented and tested in full, would be evaluated based on two primary criteria: improvements in the quality of machined parts and reductions in overall production time. Improvements in machining quality would be assessed using a Coordinate Measuring Machine (CMM) to compare the dimensional accuracy of parts produced with and without compensation applied on the hybrid mill. Specifically, the end-cap component would be measured to determine whether the system could reduce dynamic machining errors during the manufacturing process. Production time would be evaluated by measuring the time required to

machine three end-cap components sequentially after the machine setup phase. This comparison would be conducted between a human operator and the COBOT-assisted process. During this procedure, the unprocessed part would be unloaded three times and loaded twice. As the machining time for each component remains constant, any difference in total production time would result from the duration required to load and unload the parts.

8.2 A digital twin-driven ultra-precision machining framework

To address this, a comprehensive framework will be developed to outline how the proposed approach n would operate within a full manufacturing workflow for the hybrid system. This framework will define the interactions between the loading, machining, and unloading stages of part production using an integrated robotic arm. It will serve as a modular structure, allowing for the eventual integration of an enhanced and fully functional digital twin once further research and development are completed. A demonstration of the framework in operation will also be presented to illustrate the practical implementation and interoperability between the hybrid mill and COBOT within a smart manufacturing environment. For this framework to apply to other machines, only the digital twin (DT) housed within the PC unit would require retraining of its machine learning model using data specific to the new machine. This frame can be seen in Figure 8-1 below.

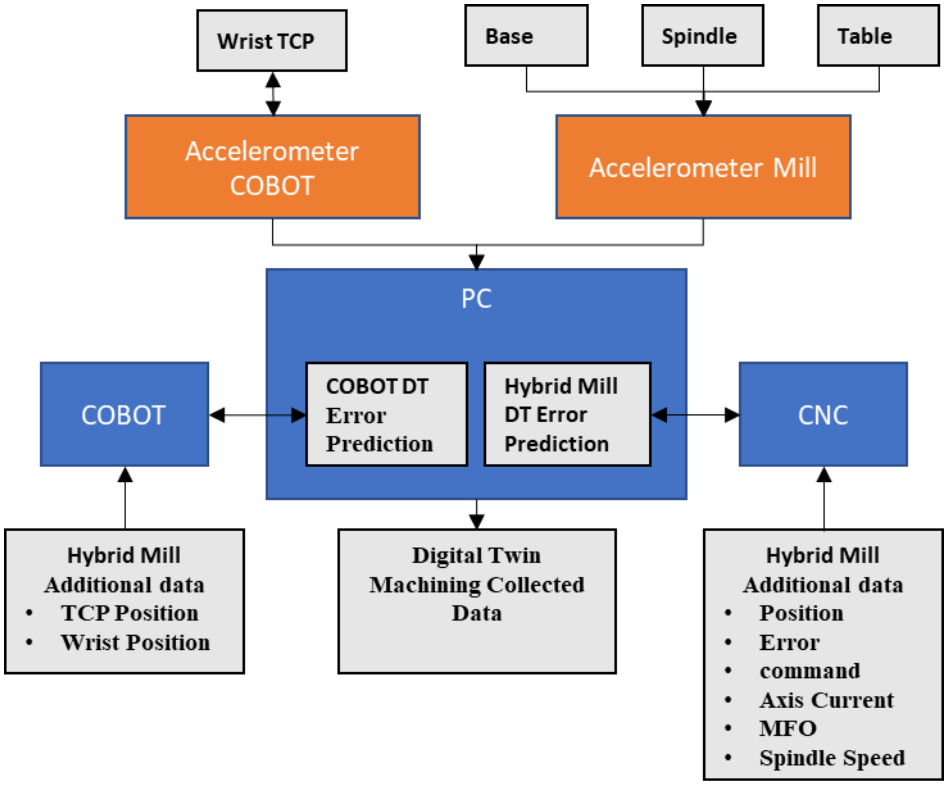


Figure 8-1 Digital twin-driven Smart manufacturing system framework

8.3 Experimental setup

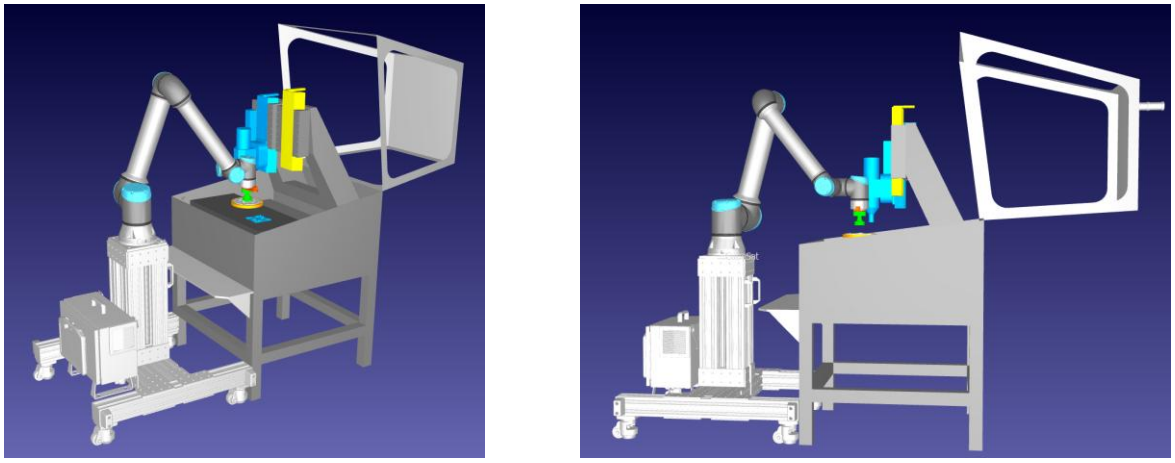
A demonstrator system of the proposed smart manufacturing system framework was constructed to provide practical insight into its real-world operation. This demonstrator performs the final machining passes on a Cube Satellite (CubeSat) end cap. The CubeSat end cap was 3D printed to enable rapid production, with an additional 1 mm of material added to the internal surfaces to allow for the finishing operations to be performed by the hybrid mill.

The loading process of the CubeSat end cap is conducted by the COBOT, which utilises the working digital twin to apply corrective actions for accurate placement within the jig. Once positioned, the COBOT holds the component securely until it is fastened in place using bolts. After securing the workpiece, the hybrid mill executes the final machining passes to achieve the desired surface finish and dimensional accuracy.

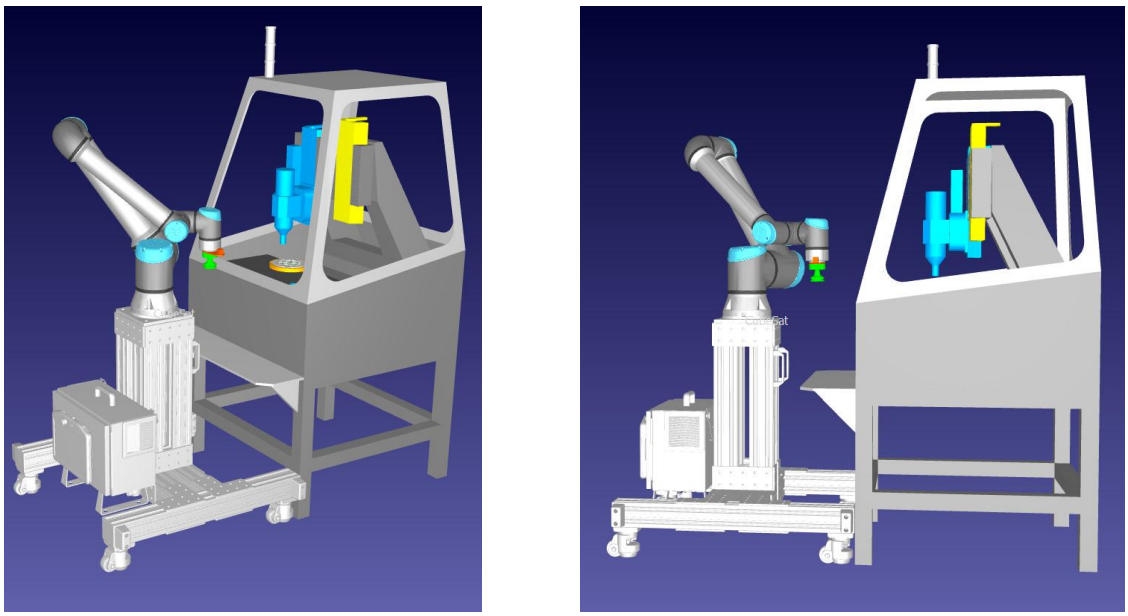
The CubeSat demonstrator was selected due to the standardised nature of CubeSat architecture, which facilitates cost-effective design, production, and deployment. A CubeSat typically measures 10 × 10 × 10 cm (1U), with larger configurations formed by combining multiple units (e.g., up to 12U). Despite this standardisation, variations in internal layouts and attached modules often lead to differences between individual builds. Consequently, a customisable and adaptable manufacturing system is required to maintain precision and cost efficiency across unique designs. The proposed smart manufacturing system aims to automate this process, providing real-time corrective actions to accommodate geometric variations while also automating loading and unloading operations. This approach would enable the production of high-quality, custom CubeSat components with improved speed, repeatability, and reduced manufacturing costs.

For this demonstrator, a modified CubeSat end frame was designed and 3D printed. The modifications allowed for easier attachment to the hybrid mill through a custom jig, as well as the inclusion of a 1 mm offset on the internal features to permit finishing by the hybrid mill. Once machining is completed, the COBOT unloads the part, allowing the process to be repeated for subsequent cycles.

The entire machining process for the CubeSat component is conducted autonomously, except for the manual securing and removal of the end cap from the hybrid mill. Full automation of this stage was not implemented due to the absence of a suitable end-effector tool for the COBOT to perform the bolting operation. The intended experimental setup is illustrated in Figure 8-2(a), showing the COBOT in operation, and in Figure 8-2(b), depicting the hybrid mill during machining.



(a)



(b)

Figure 8-2 Digital model of COBOT and hybrid mill (a) COBOT loading/unloading (b)Hybrid mill in operation

The CubeSat end cap design used in this demonstrator was derived from an open-source model, which was subsequently modified to facilitate easier 3D printing and the integration of the required jigs and fixtures. Additional design adjustments were made to ensure a suitable gripping interface for the COBOT, enabling reliable handling during both the loading and unloading operations. The final modified design, which was 3D printed and used for testing, is shown in Figure 8-3 with a technical drawing shown in Appendix 15.

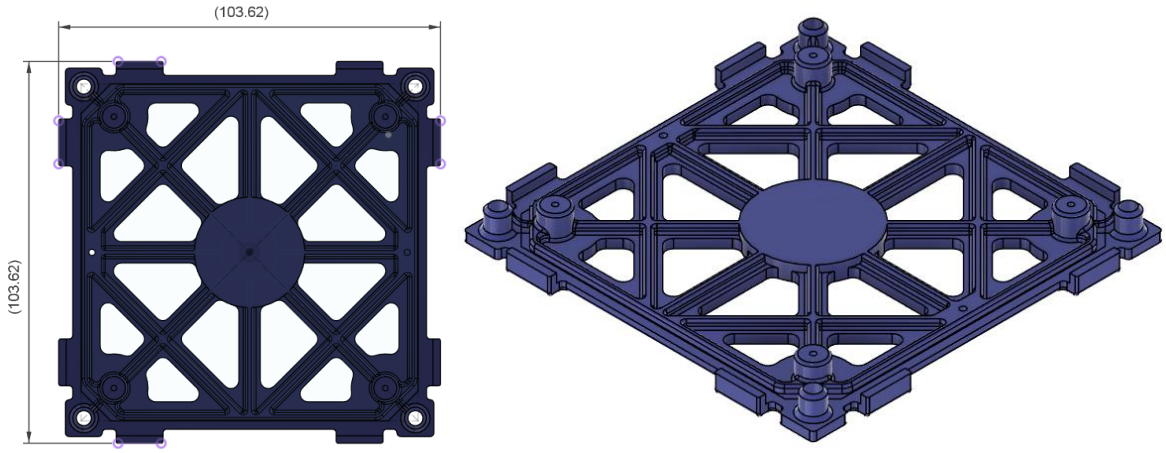


Figure 8-3 Cube Sat CAD Rendering

8.3.1 Cube sat and mounts design.

As stated previously, the CubeSat end cap will be 3D printed to facilitate rapid production and to simplify the subsequent machining process. During printing, the wall thickness of the component will be set to 2 mm, ensuring that the regions intended for machining are printed as solid sections. This approach provides material consistency and stability during the cutting process. The 3D printed component is shown in Appendix V.

The jig used for attaching the printed CubeSat parts allows the end cap to be positioned at the centre and secured using four holes with recessed nuts, enabling the end cap to be bolted in place once the COBOT positions and holds it. The jigs have been designed with slight tolerances to allow the corrective action of the digital twin to take effect. The design and dimensions of the jig can be seen in Figure 8-4, with a technical drawing shown in Appendix 15.

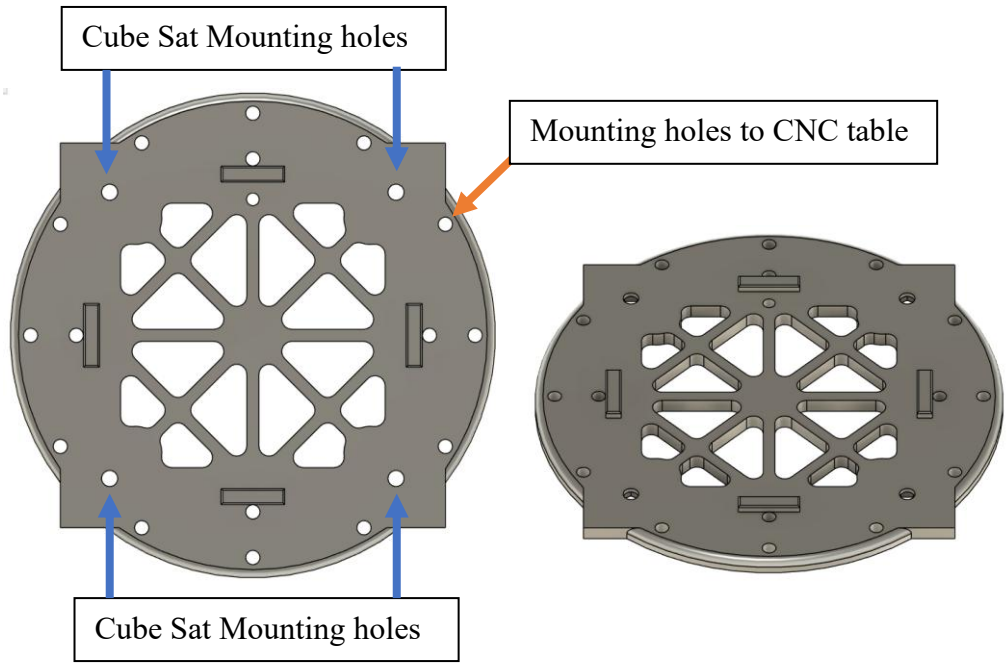


Figure 8-4 Cube sat hybrid mill table mount CAD rendering.

The COBOT manipulates the CubeSat components using a custom-designed vacuum gripper. This gripper is controlled via the COBOT's I/O interface, which is connected to a relay regulating the supply of pressurised air at 2 bars to a vacuum generator. The generated vacuum is then transmitted to the gripper nozzle, enabling secure handling of the parts during both loading and unloading operations. The dimensions of the vacuum gripper attachment are shown in Figure 8-5, with the CAD rendering presented in Figure 8-6, while the complete experimental setup under test conditions is shown in Figure 8-7.

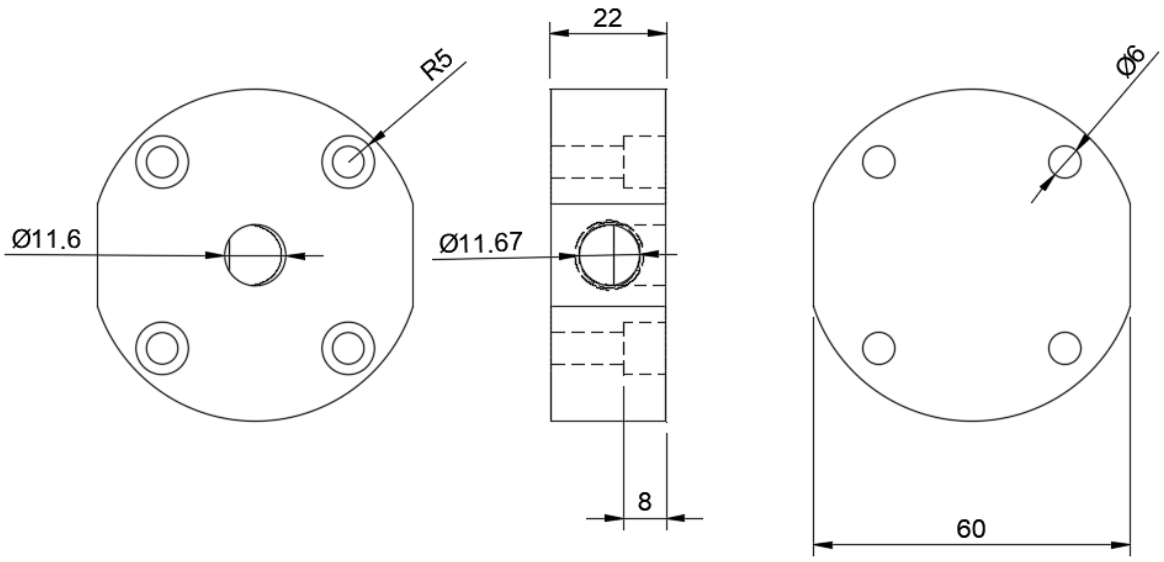


Figure 8-5 COBOT vacuum gripper engineering drawing

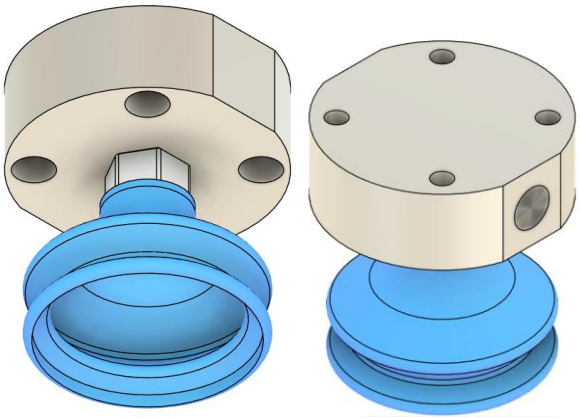


Figure 8-6 COBOT vacuum gripper CAD rendering

Figure 8-7 COBOT vacuum gripper attached

8.3.2 COBOT work-handling programmed path

The programmed motion of the COBOT for the loading and unloading operations was developed using RoboDK. Corrective action from the digital twin is applied only to the final placement target, designated as “CubeSat Place”, since the unloading motion does not require positional correction. The “CubeSat Place” target directly determines the relative position of the CubeSat end cap with respect to the hybrid mill’s origin, making it the only position where corrective adjustment is necessary and therefore evaluated.

To secure the CubeSat end cap in place, the COBOT maintains its hold on the component while the operator manually inserts and tightens the bolts. This is achieved using a pause function, during which the COBOT holds the part stationary for 25 seconds. While this method is appropriate for the present demonstrator, it would need to be replaced by a fully automated securing mechanism in an industrial implementation. Potential solutions include the use of vacuum pads, electromagnets, or an actuated vice to enable fully autonomous operation.

The RoboDK programmed simulation and corresponding target locations are illustrated in Figure 8-8 and detailed in Table 8-1.

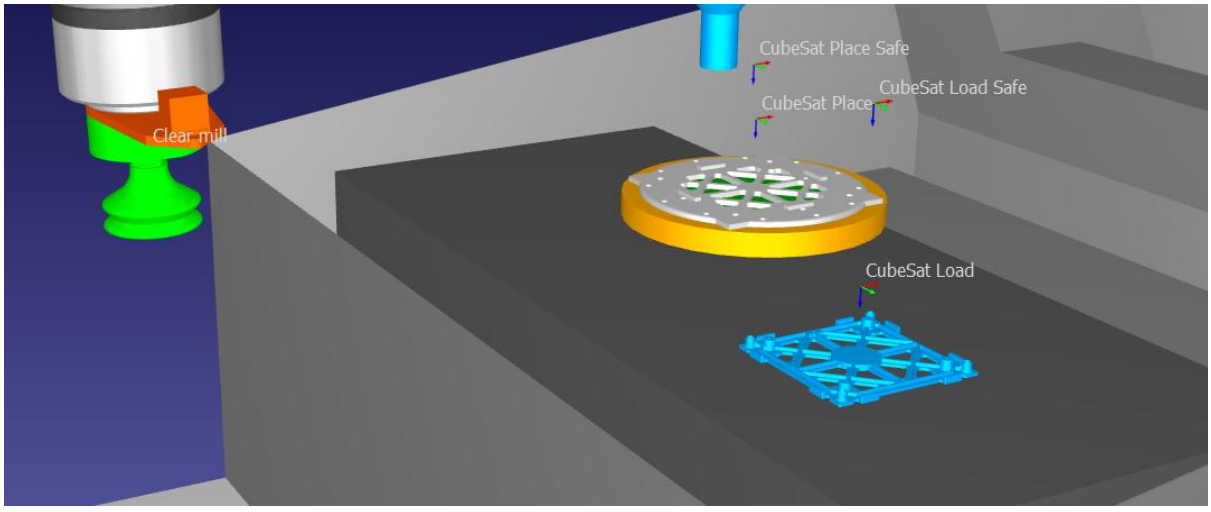


Figure 8-8 Cube sat loading targets RoboDK

Table 8-1 Cube sat target locations for loading and unloading

Target Name	X, Y, Z Location	Reference	Description
CubeSat Load Safe	-642.966, 164.140, 199.253	UR10e Base	A safe location for the COBOT to line up correctly to attach to the end cap
CubeSat Load	-664.047, 163.062, 89.076	UR10e Base	The location to pick up the end cap part
CubeSat Place Safe	-676.419, -0.746, 184.281	UR10e Base	A safe location for the COBOT to line up correctly for the placement of the end cap
CubeSat Place	-680.035, -0.756, 147.350	UR10e Base	The location of the end cap on the hybrid mill
Clear mill	-293.299, -11.747, 236.222	UR10e Base	A safe location for the COBOT so the hybrid mill enclosure can be shut for safe operation.

The process of loading the CubeSat part is shown from Figure 8-9 to Figure 8-12 at each stage in the simulation, with the part being attached using a vacuum gripper controller by the I/O pin 0 to a solenoid supplying air to the vacuum generator.

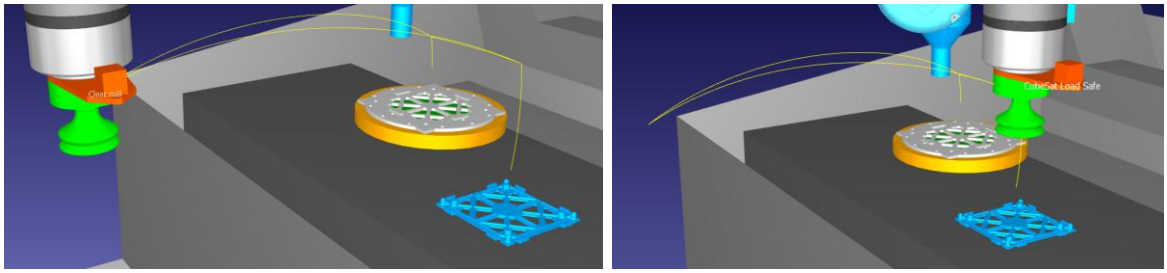


Figure 8-9 COBOT cube sat clear mill to load safe

Moves to CubeSat Load Safe for a clear approach to the end cap

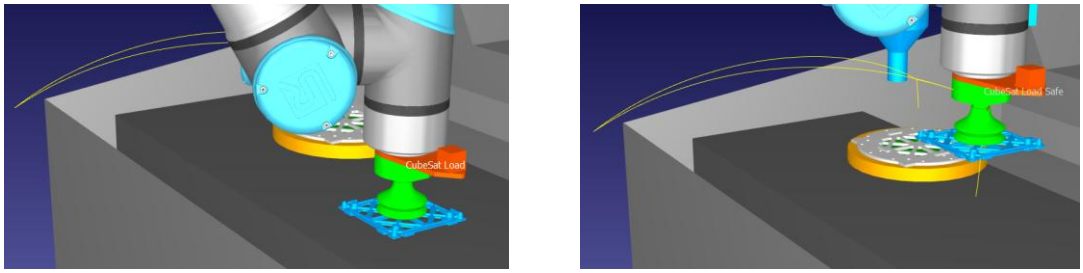


Figure 8-10 COBOT cube sat load safe to load

Moves back to CubeSat Load Safe with end cap attached

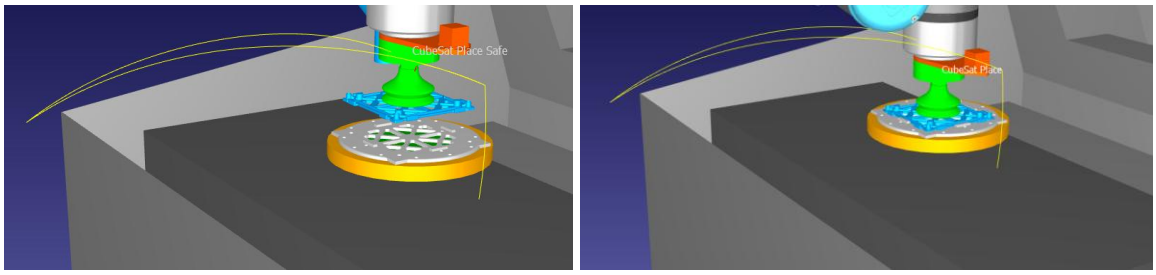


Figure 8-11 COBOT cube sat place safe to place

Moves to CubeSat Place and will drop the end cap

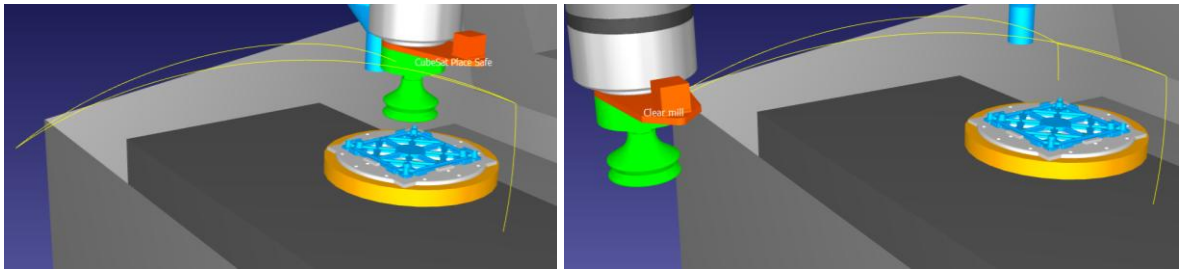


Figure 8-12 COBOT cube safe place safe to clear mill

Moves back to CubeSat Place Safe, leaving the end cap on the hybrid mill

The code that is being used for this process is shown below with an explanation for each section in Figure 8-13 for loading and Figure 8-14 for unloading.

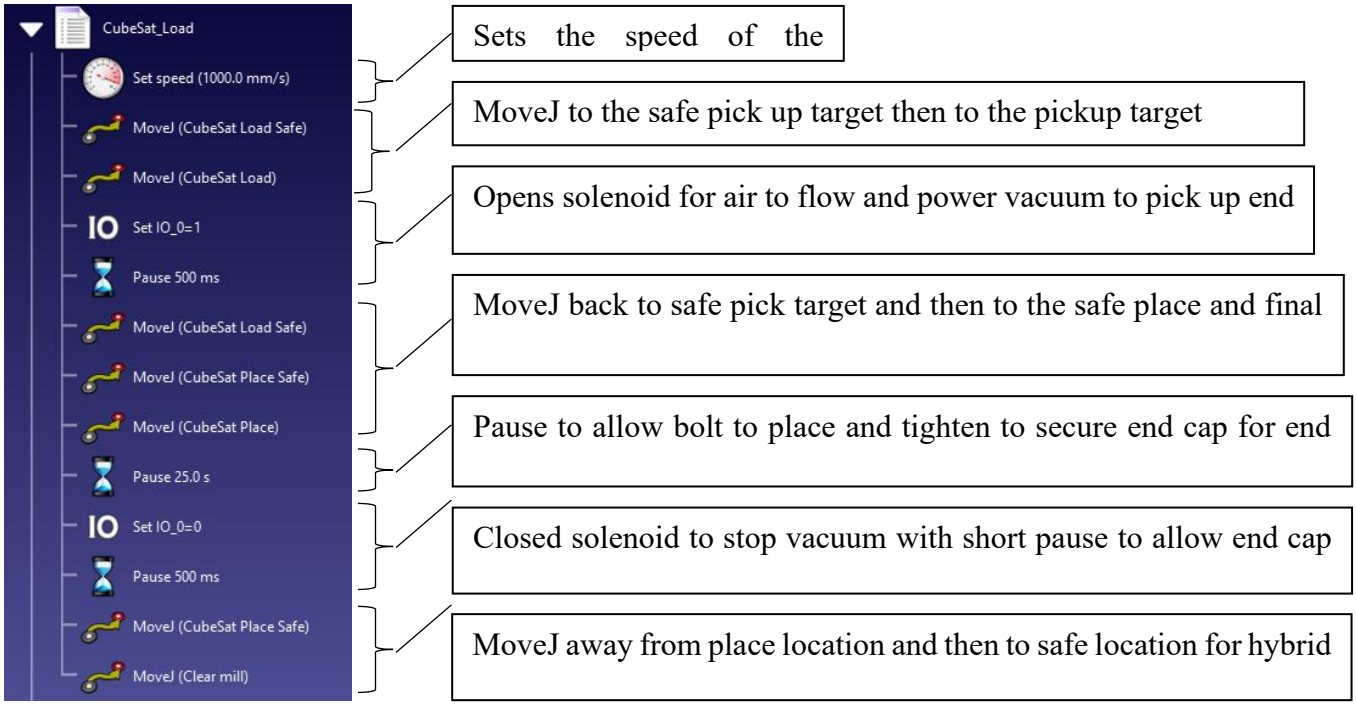


Figure 8-13 Cube sat load program

As previously stated, the unload program does not require corrective action in this demonstrator, as the final placement position is not critical to the overall process. The programmed sequence for the unloading operation, shown in Figure 8-14, functions similarly to the loading routine but in reverse. During this operation, the COBOT picks up the CubeSat end cap from the placement location and returns it to the start position defined in the loading program.

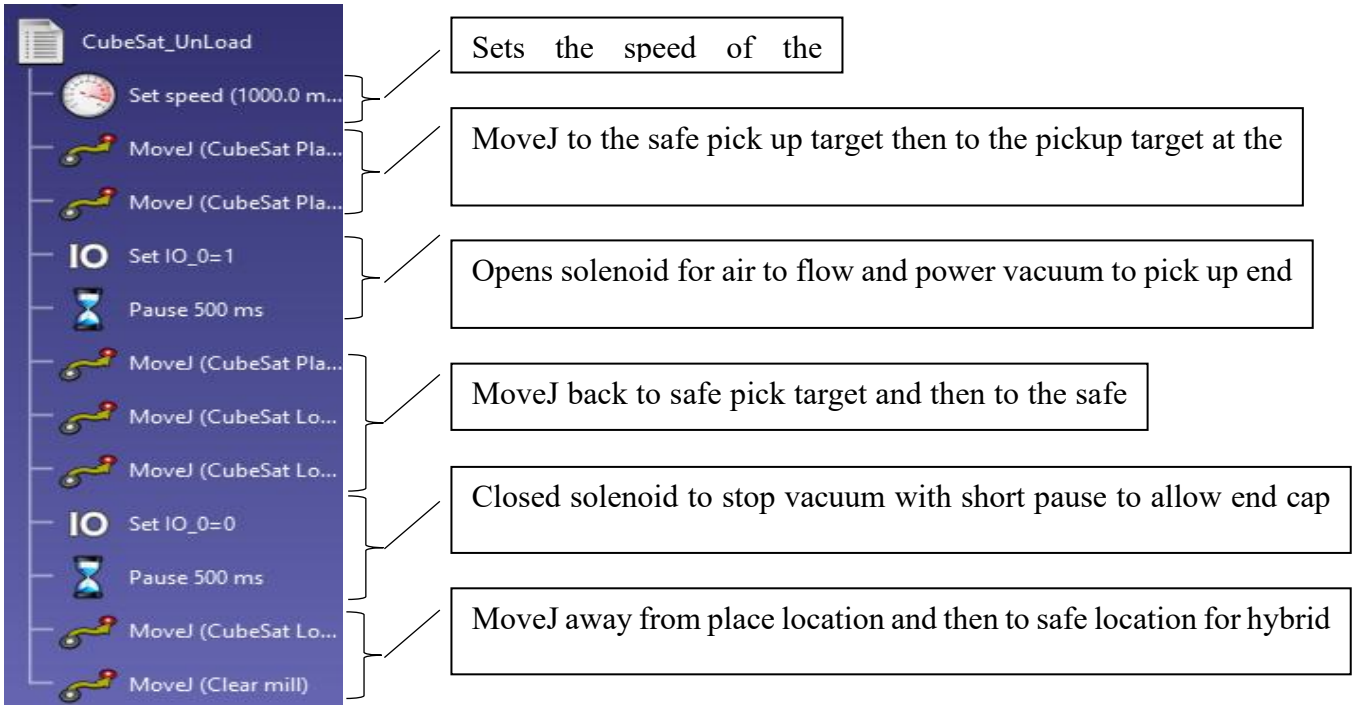


Figure 8-14 Cube sat unload program

8.3.3 Hybrid mill G-code programming

The final stage of the demonstrator involves the G-code for the hybrid mill. This code has been generated to remove the additional 1 mm offset from the internal frame sections of the CubeSat end cap. Due to the limited build volume of the hybrid mill, the machine must move to a predetermined load/unload position, after which the end cap is rotated about the C-axis to allow each quarter of the part to be machined sequentially. The G-code used for one section of the CubeSat end cap is shown in Figure 8-15.

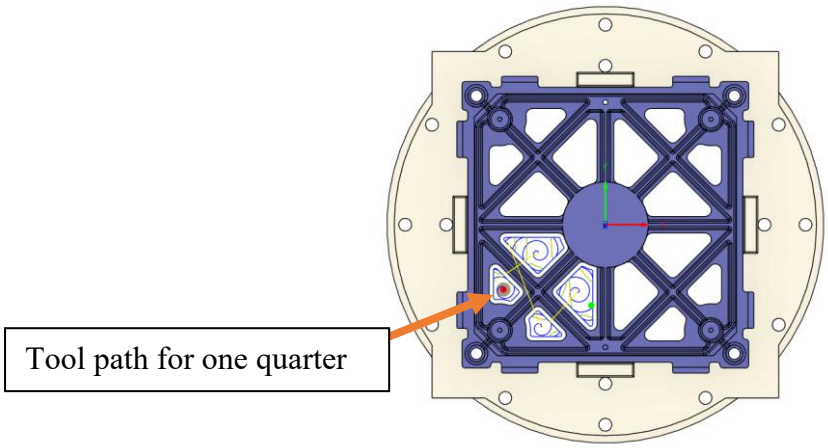


Figure 8-15 Cube sat machining tool path

The machining of each quarter, as previously stated, is achieved through 90-degree rotations of the C-axis. Since the end cap is centrally positioned within the jig, it can be rotated precisely, allowing the same G-code to be reused for each quadrant. The movement to the load/unload position occurs at both the beginning and end of the machining program, with the position defined as X = –114.65 mm, Y = 0 mm, and Z = 80 mm. This ensures a consistent reference point for loading and unloading operations while also positioning the spindle at a known safe location during the machining process. The overall process of machining each quarter is illustrated in Figure 8-16.

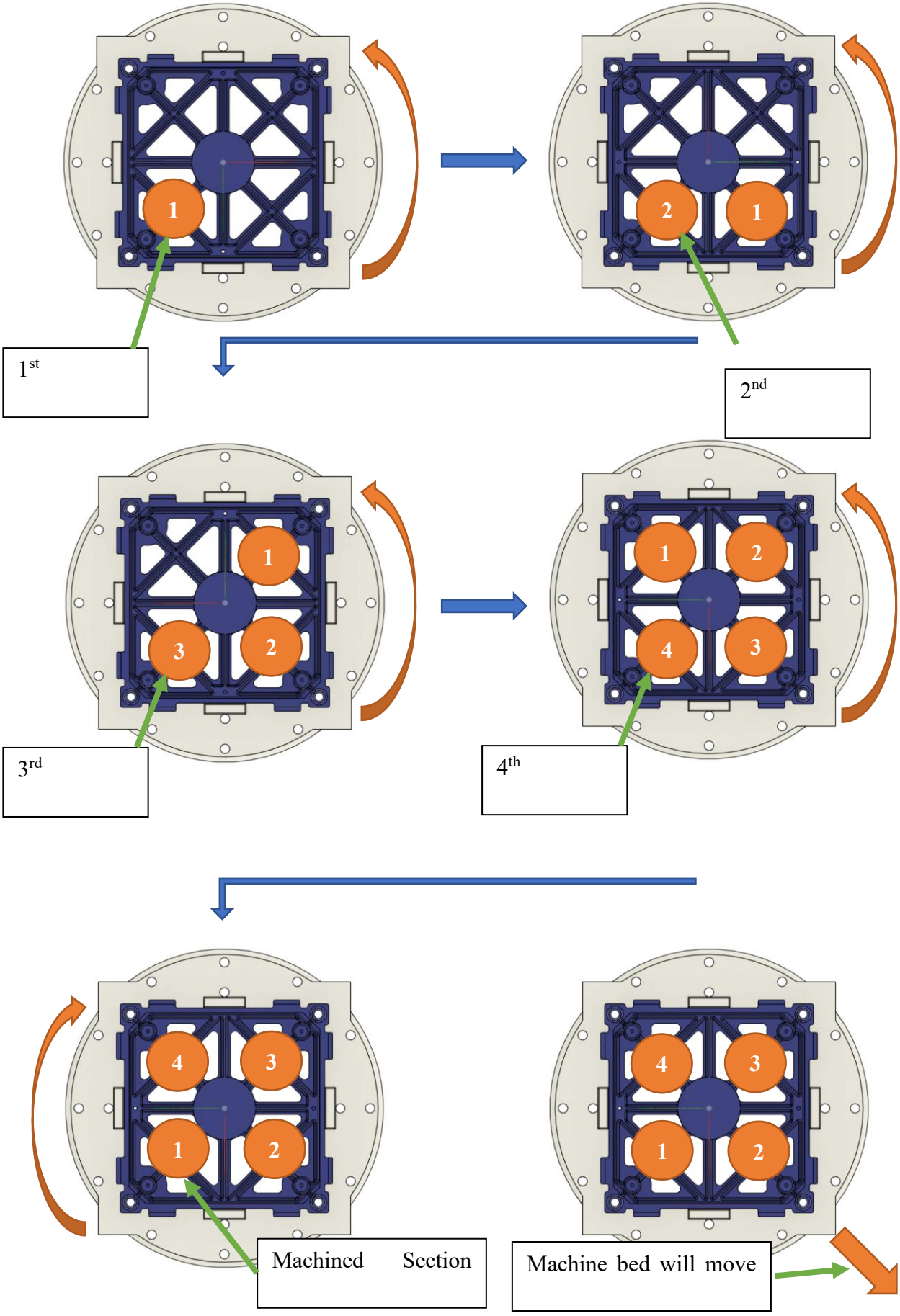


Figure 8-16 Cube sat complete machining process

As the digital twin for corrective action on the hybrid mill shown in Chapter 6 was not yet operating with sufficient reliability, it was not implemented in the demonstrator at this stage. However, a functioning version of this system could be integrated into future work, as outlined in the proposed framework. Despite this limitation, the digital twin for the smart manufacturing system continued to collect relevant data from the hybrid mill, which could later be used for predictive modelling. This also served to verify that the system architecture is capable of handling the computational load required to operate the digital twin in real time.

8.4 Operation of a digital twin-driven smart ultra-precision machining system

With the demonstrator fully defined, it now needs to be executed using the available equipment to verify that the system can perform in a manner consistent with the proposed smart manufacturing system framework. Once the CubeSat end caps have been 3D printed, they can be positioned on the hybrid mill to enable the COBOT to complete the pick-and-place operation before machining. An example of one of the printed CubeSat end caps is shown in Appendix V.

8.4.1 Loading Stage

The loading process was successful, with the COBOT reliably and repeatedly able to load the CubeSat end cap into the hybrid mill for machining. This demonstrates the accuracy and consistency of the programmed motion and vacuum gripping system. The complete loading sequence is illustrated in Figure 8-17, showing the process as defined and simulated in RoboDK.

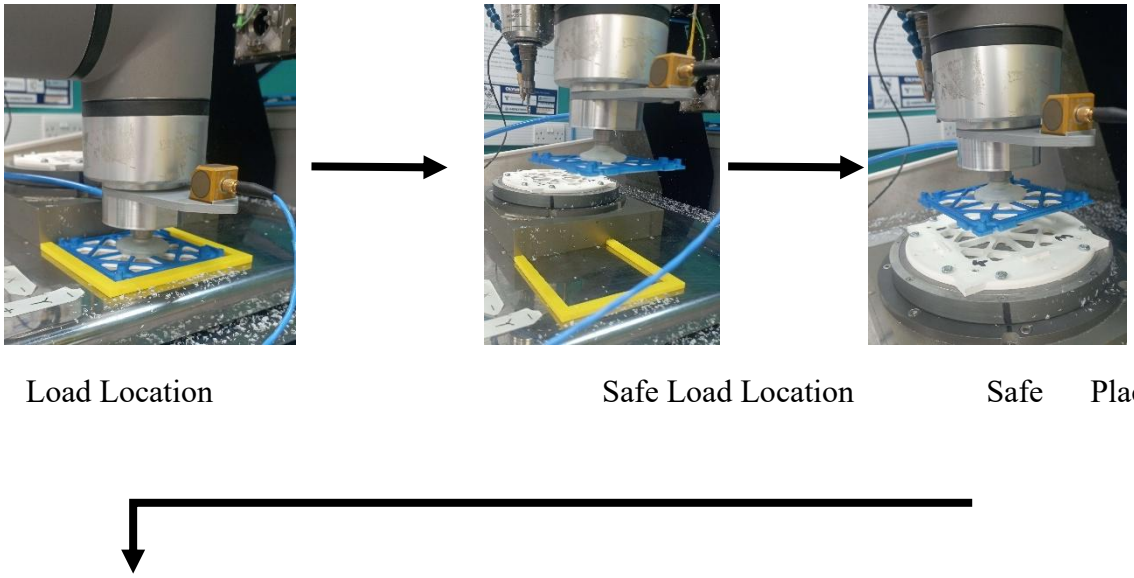


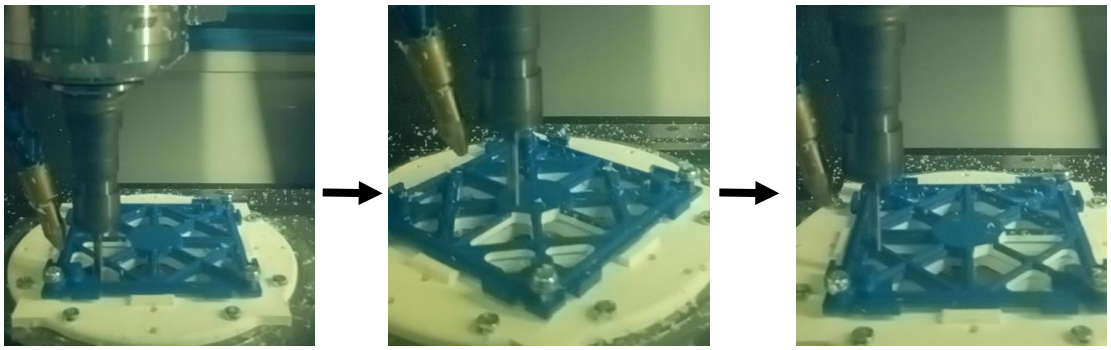
Figure 8-17 Complete loading process of CubeSat using COBOT

Once the end cap was securely fastened onto the hybrid mill and the COBOT had moved away, allowing the safety enclosure to close, the machining process commenced. During this stage,

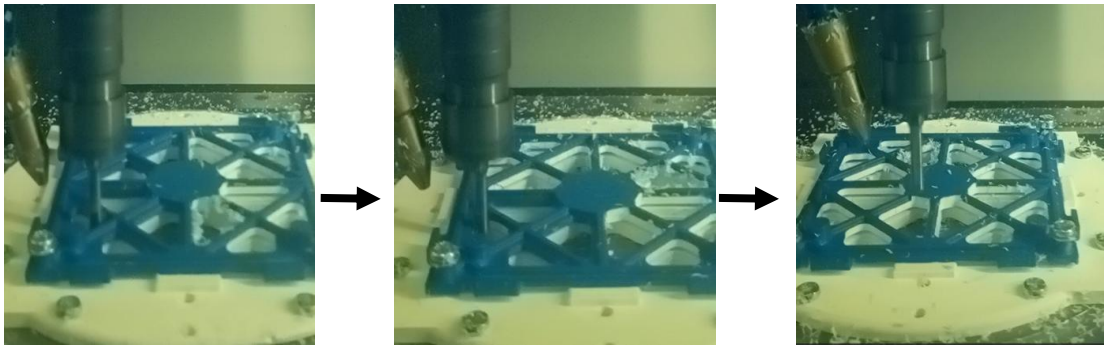
the hybrid mill successfully removed the 1 mm offset that had been intentionally applied to the internal sections of the 3D-printed end cap, bringing the part to its final design dimensions.

8.4.2 Machining Stage

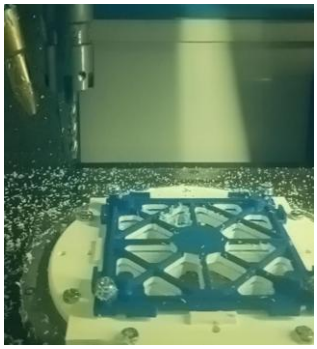
With the end cap securely positioned in the hybrid mill, the machining stage of the digital twin process could begin. Once the enclosure door was closed, the machining program was initiated, executing the G-code to machine the first quarter of the part. The process then continued sequentially until all four quarters were completed, after which the hybrid mill automatically returned to the load/unload position. The complete machining sequence is illustrated in Figure 8-18.



1st Quarter Machining Rotates to Next Quarter 2nd



3rd Quarter Machining 4th Quarter Machining Rotates Back to



Moves to Load/Unload

Figure 8-18 Complete machining process of the cube sat in the hybrid mill.

With the machining stage complete, the part has been machined to final dimensions and will now need to be unloaded from the hybrid mill, with the full process being shown in Figure 8-19.

8.4.3 Unloading

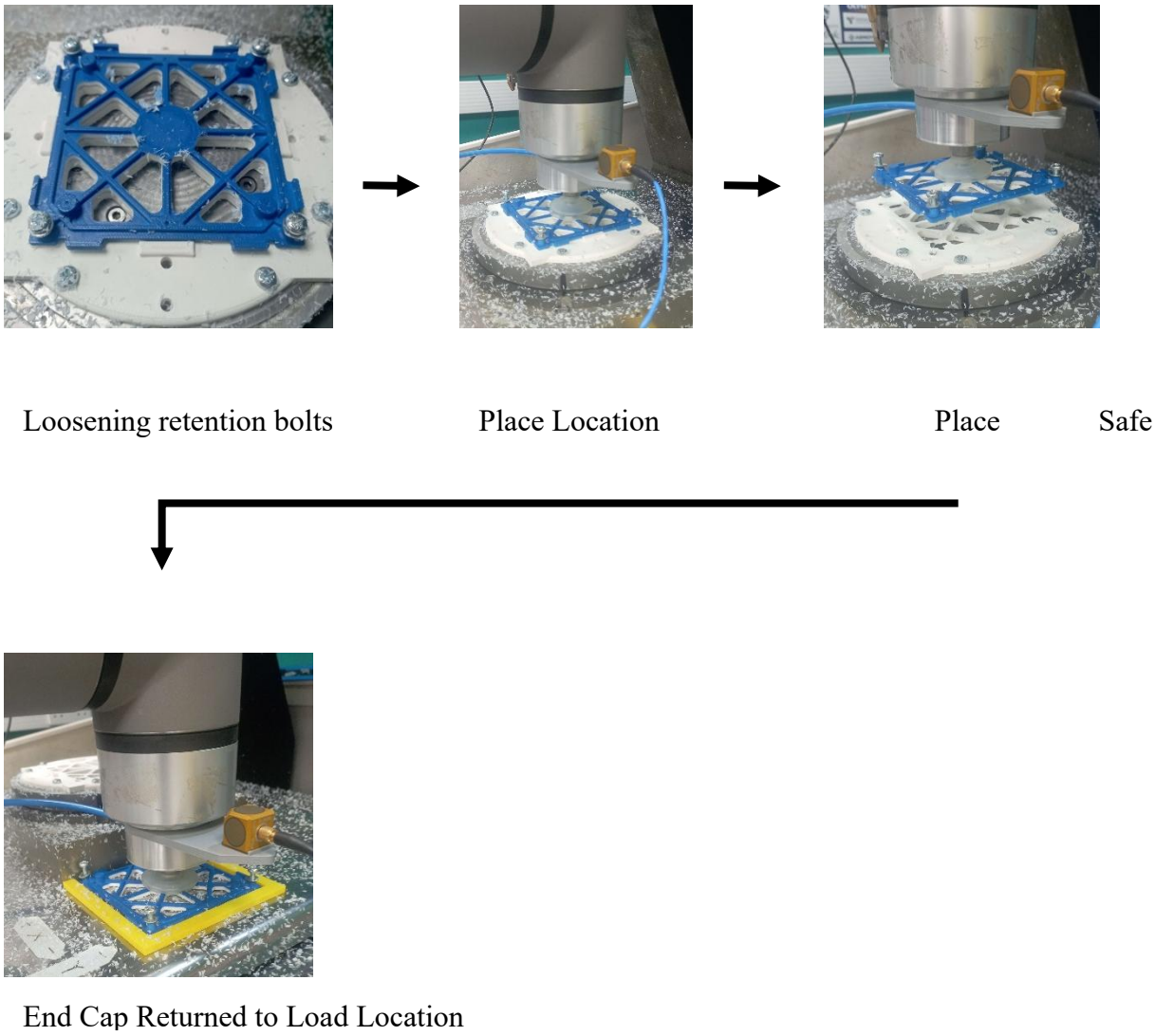


Figure 8-19 Complete Unloading process of the cube sat using COBOT.

Upon completion of the machining stage, the end cap had been machined to its final dimensions and was ready to be combined with the rest of the cube's body, as shown in Figure 8-20.

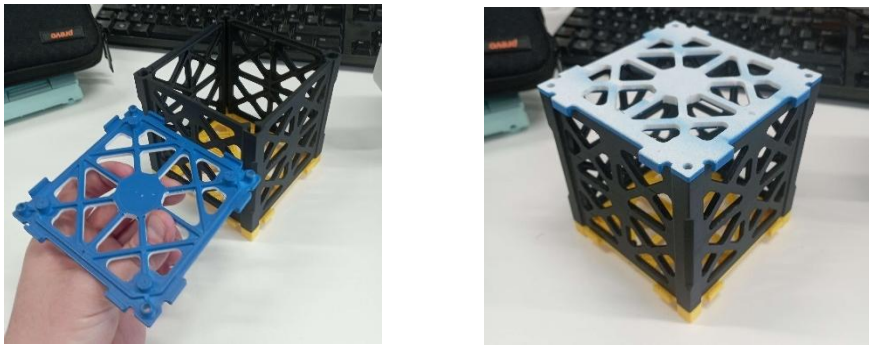


Figure 8-20 Cube sat end cap with complete frame

8.5 Benefits of a digital twin-driven ultra-precision machining system

The final demonstrator, based on the machining of a CubeSat end cap, indicates that the proposed system could enable a reduction in production time with relatively minor modifications. These modifications would primarily involve the integration of an automated clamping mechanism, removing the need for operator intervention during part fixturing.

The introduction of automated clamping would enable the possibility of lights-out manufacturing, allowing for increased operational time and, consequently, higher overall productivity. By extending machine utilisation beyond normal operating hours, more components could be manufactured within a given time period without additional labour requirements.

This approach also enables more flexible manufacturing, as setup times for new tasks could be significantly reduced. The COBOT would be capable of placing parts in the most appropriate location and orientation for each machining operation, reducing manual alignment and setup effort. While this capability demonstrates strong potential, further development and validation are required before it can be applied reliably in an industrial ultra-precision manufacturing environment.

8.6 Summary

This chapter presented the development and demonstration of a digital twin-driven smart manufacturing framework designed to integrate a COBOT and a CNC hybrid mill for ultra-precision manufacturing applications. The primary goal was to enable automated loading and unloading of components due to not having a reliable digital twin for the hybrid mill. perform machining with real-time corrective capability and achieve a higher level of autonomy within an interconnected manufacturing environment.

A demonstrator system was successfully implemented using a 3D-printed CubeSat endcap as the test component. The COBOT, equipped with a vacuum gripper and controlled through its digital twin, performed automated loading and unloading operations. The digital twin for the COBOT applied corrective actions in real time to ensure accurate placement. Once the part was secured, the hybrid mill performed the machining operations to remove the 1 mm offset from the printed model, achieving the final dimensional accuracy required.

Although the digital twin for the hybrid mill was not fully operational for dynamic error correction, the system architecture successfully facilitated machine data collection, hybrid mill control, and seamless communication between the two digital twins. The demonstrator verified that the overall framework is capable of coordinating multi-machine processes while maintaining process data flow and system responsiveness, providing a strong foundation for future implementation of closed-loop corrective control.

The results demonstrated that automated handling using the COBOT's digital twin could significantly improve precision and process repeatability, contributing toward higher productivity and reduced operator dependency. However, further development is required to improve the hybrid mill's digital twin to achieve consistent real-time corrective action,

particularly through improvements in sensor resolution, data acquisition frequency, and model retraining using expanded datasets.

All in all, the research presented in this chapter has shown that the proposed digital twin framework provides a viable proof-of-concept for a smart manufacturing system capable of integrating robotics and ultra-precision machining. While not yet fully autonomous, the system establishes a scalable platform for continued research into digital twin coupling, sensor reliability, and machine learning-driven corrective control. Future work will focus on enhancing the predictive accuracy of the hybrid mill's digital twin and implementing full automation of the securing and release operations to achieve a complete closed-loop smart manufacturing system.

Regarding the research objectives set at the start of the thesis, this research developed a digital twin architecture capable of integrating robotic workpiece handling and CNC machining within a smart manufacturing framework. The digital twin for the COBOT was successfully implemented and demonstrated the ability to apply real-time corrective actions during automated loading and unloading operations. This work establishes a method by which predictive digital twins can be used as an external feedback layer to modify machine commands and compensate for predicted dynamic errors during operation. Although a predictive digital twin for the hybrid mill was developed, its real-time corrective capability could not be fully realised within the scope of this research due to sensing and system stability limitations. Nevertheless, the system architecture successfully enabled machine communication, data collection, and coordinated operation between the machines, demonstrating the feasibility of a multi-machine digital twin framework to try to achieve research objective three.

Practical validation was carried out through the development of a demonstrator manufacturing system integrating a COBOT and a CNC hybrid milling machine. A CubeSat endcap component was used as the test case to demonstrate automated workpiece handling and machining operations within the proposed digital twin framework. The experimental results show that the digital twin-driven correction applied to the COBOT improved placement accuracy and process repeatability during automated handling tasks. While full dynamic error correction for the hybrid mill was not achieved, the system successfully demonstrated coordinated machine control, data exchange, and automated process execution to try to achieve objective four.

9 Chapter 9 – Conclusion and future work

9.1 Conclusion

This research developed a real-time, digital twin–driven control approach for ultra-precision machining and workpiece handling to correct dynamic errors. Dynamic errors were measured outside the control loop using accelerometers, with displacement calculated through double integration and later machine learning. The method was tested on a UR10e COBOT and a custom-built hybrid mill, and real-time corrective actions were applied to improve machine control. For the COBOT, this resulted in a tracking improvement of 41.3%, demonstrating that accelerometer-based dynamic error measurement can improve position tracking.

Building on this, explainable artificial intelligence (XAI) was investigated to reduce computational load by predicting displacement directly from unfiltered acceleration signals, removing the need for FFT filtering and double integration. Using QLattice, empirical equations were generated from ball bar training data across multiple feed rates, using ten extracted features per axis. The trained XAI models were compared with black-box methods and applied for corrective action, achieving error reductions of 31.5% MoveJ and 4% MoveL in COBOT pick-and-place tests, while the hybrid mill showed limited improvement, with the best average improvement being -6.6%. The reduced hybrid mill performance is believed to be influenced by accelerometer frequency limits, sensor sensitivity, and environmental effects, requiring further investigation.

From this, it is shown that XAI can establish a model that can predict the accuracy of the dynamic errors of a COBOT and correct for this with high computational efficiency.

A proof-of-concept demonstrator was also completed using the COBOT to load and unload a CubeSat end cap for final machining on the hybrid mill, showing coordinated operation and

the potential for future automated ultra-precision manufacturing with real-time corrective action.

9.2 Contribution to knowledge

Objective 1 – Understanding digital twins for ultra-precision manufacturing

This research provided a critical review of state-of-the-art digital twin technologies within advanced manufacturing and identified key knowledge gaps in their application to ultra-precision machining. In particular, the research highlighted the lack of effective methods for measuring and compensating for machine dynamic errors occurring outside the machine control loop, establishing the need for external sensing and predictive modelling approaches.

Objective 2 – Measurement of dynamic errors outside encoder feedback systems

A novel measurement method using accelerometers was developed to capture machine dynamic errors that are not detectable through standard encoder feedback. The approach estimates displacement from acceleration data and enables real-time monitoring of machine dynamic behaviour. Experimental results demonstrated successful application on the COBOT and showed potential for the hybrid mill, indicating that the method may provide a transferable sensing approach for other CNC machines and robotic systems.

Objective 3 – Development of a digital twin-driven dynamic error compensation approach

An explainable artificial intelligence (XAI)-based predictive digital twin was developed to estimate dynamic errors from acceleration signal features. The model generates interpretable mathematical expressions that enable fast real-time predictions with reduced computational requirements. This predictive capability was integrated into a digital twin framework to apply corrective actions during machine operation, providing an external error compensation mechanism outside the machine control loop.

Objective 4 – Experimental validation of the proposed approach

The effectiveness of the proposed framework was validated through practical testing using a COBOT and a CNC hybrid milling machine. The results demonstrated that the digital twin-driven compensation approach improved positioning accuracy in robotic workpiece handling tasks and showed potential for application in machining systems. Although further development is required to achieve consistent dynamic error correction for ultra-precision machining, the experimental results confirm the feasibility of the proposed approach.

This research highlights several areas where further investigation is required to advance the proposed approach. Future research should therefore focus on the following areas:

Further investigation into machine learning and XAI models

Additional research should explore alternative machine learning and explainable AI (XAI) models to develop a more reliable prediction framework. The current model demonstrates inconsistencies at certain feed rates, which indicates the need for improved model robustness. This may involve collecting larger datasets, evaluating different modelling techniques, or incorporating additional data sources to improve prediction accuracy and stability.

Investigation into the limitations of the dynamic error measurement approach

Further research is required to determine the limitations of this novel accelerometer-based method for measuring dynamic errors, particularly for ultra-precision machining systems such as the hybrid mill. Improving the accuracy and reliability of error prediction may require the use of higher-sensitivity accelerometers, improved data acquisition systems, alternative modelling approaches, or expanded training datasets to better capture the dynamic behaviour of the hybrid milling machine.

Validation of the complete digital twin-driven manufacturing system

Future work should focus on implementing and testing the complete system within a real manufacturing scenario. This would require both the COBOT and hybrid mill digital twins to operate reliably so that the higher-level coordinating digital twin can control the full manufacturing process. Experimental evaluation should determine whether the proposed system improves production quality and reduces production time compared with manual operation. Machined parts could be assessed using standard quality control techniques, such as coordinate measuring machine (CMM) inspection, to provide quantitative measurements of dimensional accuracy. Production efficiency could be evaluated by comparing the time required to manufacture batches of components under automated and manual handling conditions, acknowledging that setup time may vary between operators.

9.3 Conducted future work

Future research should focus on improving the reliability of the hybrid mill digital twin to enable a complete evaluation of a digital twin–driven ultra-precision machining system. With a fully developed system, it would be possible to conduct a more comprehensive investigation into the effectiveness of using explainable artificial intelligence (XAI) and acceleration-based measurements to detect and correct dynamic machine errors as a universal approach. The proposed system has the potential to introduce a paradigm shift in manufacturing by providing a low-cost method of measuring machine error independently of the machine controller. Such an approach could be integrated into a wide range of manufacturing systems. This may lead to a reduction in the cost of precision components, as machining to tight tolerances would become more reliable. In addition, production times could potentially be reduced if higher feed rates could be used while maintaining the same tolerance requirements. To achieve this, further investigation is required to understand why the proposed approach demonstrated improved performance for the COBOT but not for the hybrid mill. Future work could explore the use of accelerometers with different frequency ranges or higher measurement sensitivity to determine

whether the resolution of the current sensors is insufficient for ultra-precision machining applications. Further research should also examine the broader applicability of this novel method for the COBOT beyond pick-and-place tasks. The results presented in this research indicate that the approach shows promise as a universal solution for other robotic systems, particularly given the similarities between collaborative robots and industrial robotic arms. This could enable improved control and error compensation across a wide range of industrial applications. Additionally, the approach may have potential applications in medical robotics, where reducing dynamic errors during robotic surgery could improve safety and enable more complex surgical procedures, ultimately enhancing patient outcomes. Finally, further research should focus on integrating the digital twin directly within the machine control architecture and sensor systems in order to reduce communication latency. Testing conducted during this research indicates that the greatest delay in error prediction resulted from communication between multiple software programs required to exchange data. Developing a unified system capable of simultaneously controlling the machine, collecting sensor data, predicting errors, and applying corrective actions would significantly reduce latency and improve the accuracy and effectiveness of the error compensation system.

9.3.1 Low-frequency accelerometers testing

The frequency limits of the accelerometers currently used should be examined. Additional data has been collected for training purposes using the PCB 3743F112G accelerometer, which has a frequency range of 0–250 Hz (±5%). While this would address the low-frequency measurement limitations, higher-frequency components would not be captured, potentially requiring supplementary solutions. The data was collected in the same manner as the previous hybrid mill data and processed using the same methods. The results of the training process are presented in Figure 9-1 and .2 for the regression learning (RL) model and in Figure 9-3 and Figure 9-4 for the explainable artificial intelligence (XAI) model.

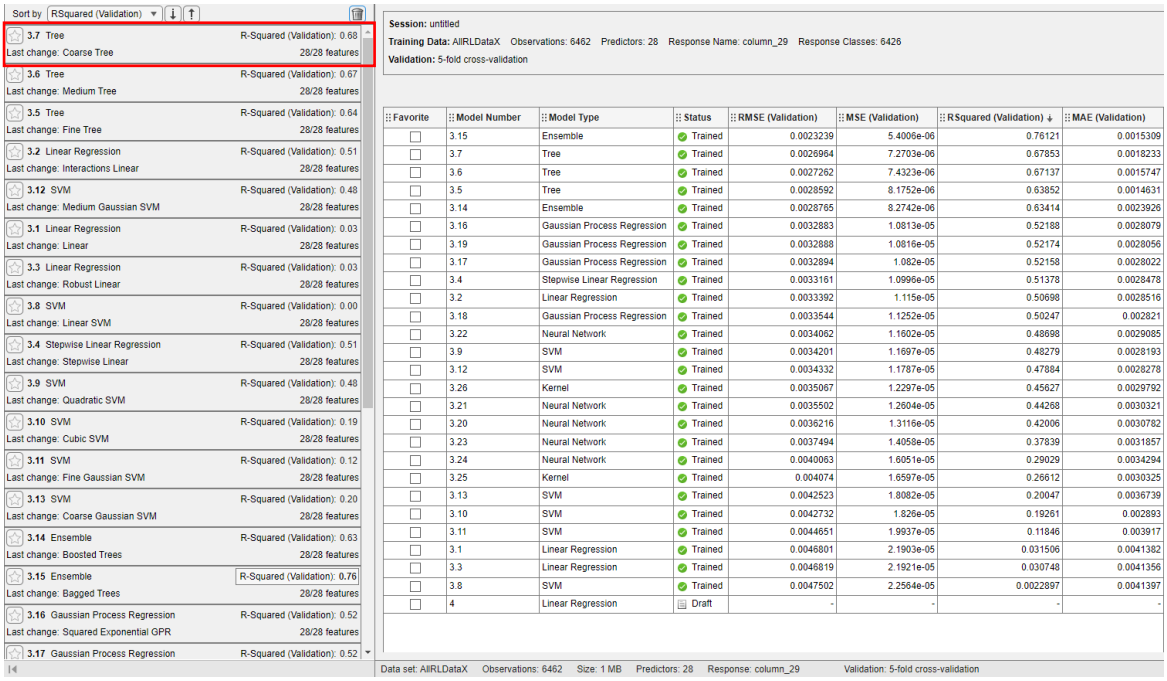


Figure 9-1 X RL training results, low frequency testing

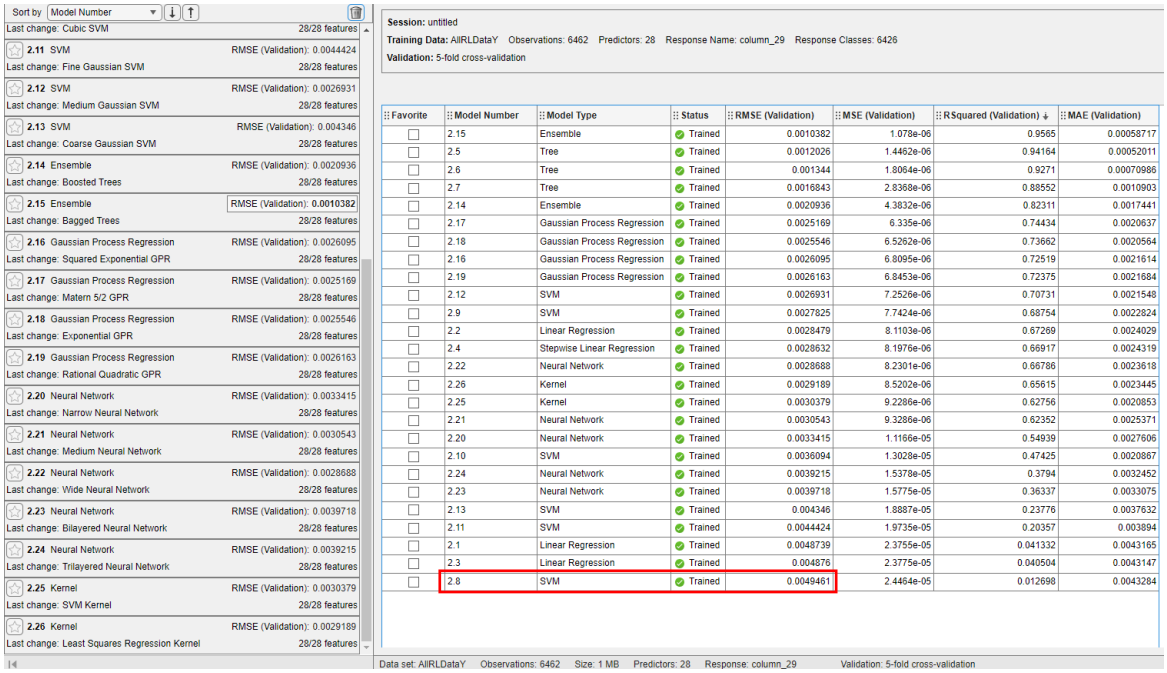


Figure 9-2 Y RL training results, low frequency testing

As can be seen, the performance of the regression learning (RL) model is relatively poor when using the lower-frequency data, achieving an R-squared value of 68% for the X-axis and only 4.9% for the Y-axis.

9.3.2 XAI Low Frequency Training Results

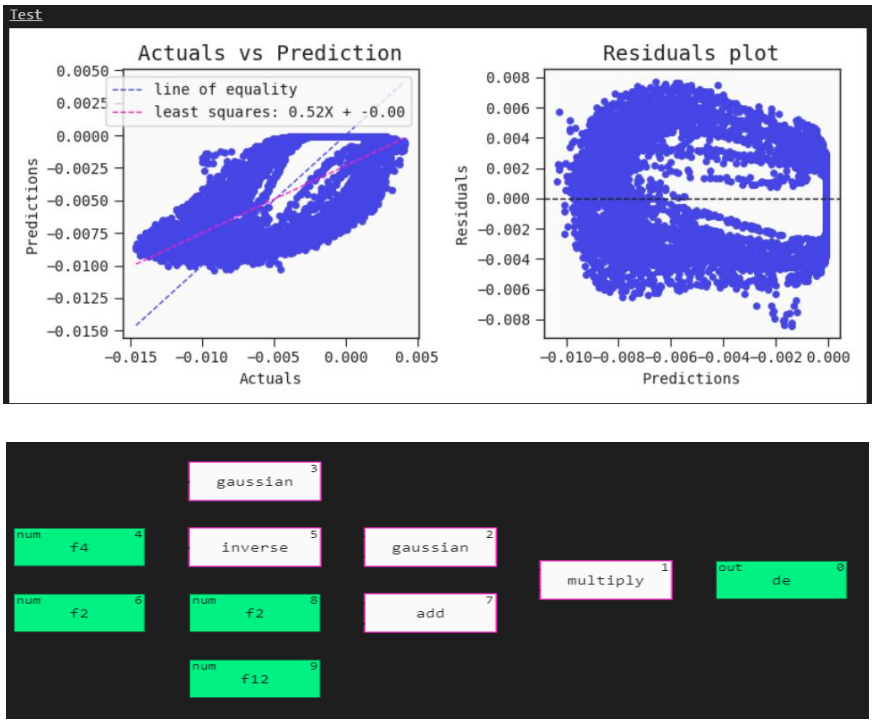


Figure 9-3 X XAI training results, low frequency testing

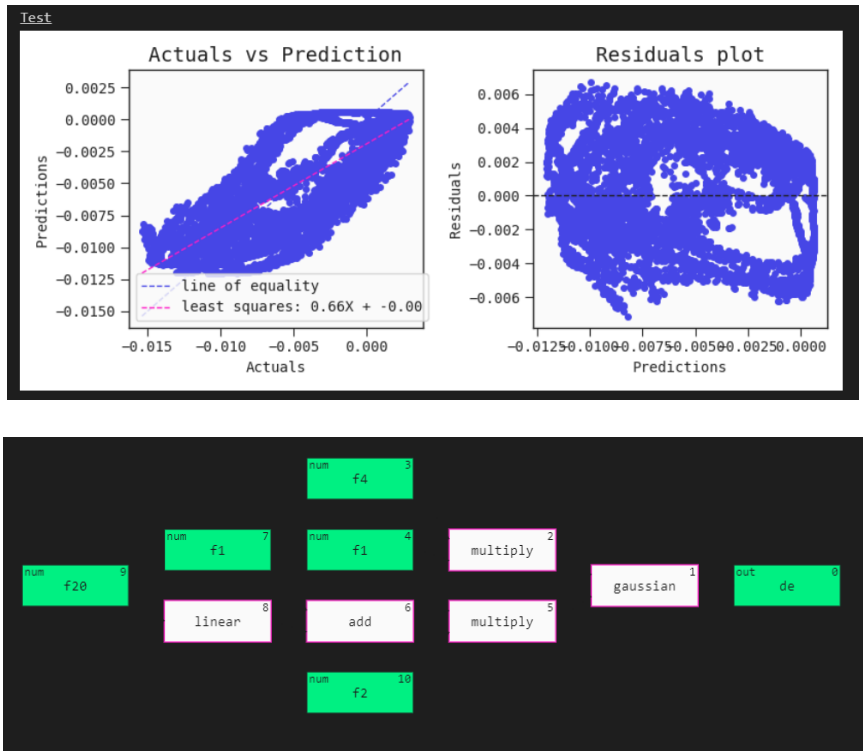


Figure 9-4 Y XAI training results, low frequency testing

As shown in the XAI results, the use of low-frequency accelerometer data did not lead to an improvement in the training performance, with the X-axis achieving an R-squared value of 52% and the Y-axis achieving 66%.

Based on these training results, it is believed that the low-frequency accelerometer is not a suitable replacement for the accelerometers previously used during testing and model training. This finding further suggests that the issue affecting the hybrid mill's training accuracy is more likely related to the lack of environmental control such as variations in temperature and humidity or the need for more sensitive accelerometers capable of capturing finer-scale dynamic behaviour.

Once these investigations are complete, the full digital twin system for controlling both machines should be tested to assess its effectiveness and identify potential issues during operation. Without a reliable digital twin solution for corrective action on the hybrid mill, the complete smart manufacturing system will remain limited in its ability to enable improvements in ultra-precision manufacturing.

9.3.3 Environmental data

During the experiments, environmental data were not considered for the hybrid mill, which may have affected the machine's performance to a greater extent than for the COBOT due to the hybrid mill's higher precision. The laboratory used for testing was not temperature-controlled, allowing variations in ambient temperature and humidity to influence the training data. As a result, the XAI model may not have performed reliably, since the time required to collect and process data and to train the model allowed environmental conditions to change. These changes would alter the dynamic behaviour of the machine, leading to inconsistencies between training and testing conditions.

Although these differences are relatively small, they are sufficient to introduce errors in the XAI model at the resolution at which the hybrid mill operates. This sensitivity highlights the importance of environmental stability when developing predictive models for ultra-precision systems.

To address this issue, two approaches are proposed. The first is to repeat the experiments in a temperature- and humidity-controlled laboratory, allowing environmental variables to be effectively isolated and ignored. The second approach is to incorporate environmental parameters, such as temperature and humidity, directly into the training data. However, this approach presents challenges, as achieving a sufficiently wide and representative range of environmental conditions is difficult without a reliable method of controlling temperature and humidity within the laboratory.

9.3.4 More sensitive accelerometers

Due to the higher accuracy of the hybrid mill compared to the COBOT, the accelerometers used in this study may only be sufficiently sensitive for operation within the error range of the COBOT and not for the hybrid mill, where dynamic errors are approximately ten times smaller. With COBOT errors averaging around 69.24 µm compared to approximately 6.72 µm for the hybrid mill, the accelerometers may lack the resolution required to capture vibration signals with sufficient precision to generate accurate training data.

The most direct solution to this limitation would be to employ more sensitive accelerometers. However, this approach may introduce additional challenges during machining, such as increased susceptibility to noise or signal saturation. It may therefore be necessary to use a range of accelerometer sensitivities to develop a system capable of operating reliably across different machines and error scales, particularly for ultra-precision manufacturing applications.

The work conducted as part of this thesis aims to improve the capabilities of ultra-precision machines for a wider range of applications. This research has focused on methods to reduce the dynamic errors present in the manufacturing process, specifically during machining and workpiece handling for automated setup. Dynamic error is reduced by applying real-time corrective actions based on error predictions of the machines while in operation. To achieve this, a digital twin of the manufacturing process has been developed, capable of predicting dynamic errors for the machines and applying corrective actions based on both historical and real-time data. This digital twin integrates the two machines, allowing for a comprehensive understanding of the machining process through improved data collection, forming a smart manufacturing system. To create a functional smart manufacturing system capable of this, specific aims, objectives, and deliverables were established. These will be reviewed to determine what has been accomplished, identify potential issues, and explore how they could be addressed through further investigation.

10 References

1. Maier, J., *Made Smarter Review* 2017.
2. Maier, J., *Industrial Digitalisation Review*. 2017.
3. Goverment, U. *Digital Twin Definition*. 2025 [cited 2025 10/12/2025]; Available from: https://www.gov.uk/government/publications/digital-twin-definition/digital-twin-official.
4. Sun, J., et al., *Tool digital twin based on knowledge embedding for precision CNC machine tools: Wear prediction for collaborative multi-tool.* Journal of Manufacturing Systems, 2025. **80**: p. 157-175.
5. Sim, B. and W. Lee, *Digital Twin Based Machining Condition Optimization for CNC Machining Center.* Int. J. Precis. Eng. Manuf.-Smart Tech., 2023. **1**(2): p. 115-123.
6. He, W., et al., *A Hybrid-Model-Based CNC Machining Trajectory Error Prediction and Compensation Method.* Electronics, 2024. **13**(6): p. 1143.
7. Qi, Q. and F. Tao, *Digital Twin and Big Data Towards Smart Manufacturing and Industry 4.0: 360 Degree Comparison.* IEEE Access, 2018. **6**: p. 3585-3593.
8. Grieves, M., *Digital Twin: Manufacturing Excellence through Virtual Factory Replication.* 2015.
9. Allam, Z. and D.S. Jones, *Future (post-COVID) digital, smart and sustainable cities in the wake of 6G: Digital twins, immersive realities and new urban economies.* Land Use Policy, 2021. **101**: p. 105201.
10. Hartmann, D. and H.V.d. Auweraer, *Digital Twins*. 2020: Springer, Cham.
11. Liu, J., et al., *A digital twin-driven approach towards traceability and dynamic control for processing quality.* Advanced Engineering Informatics, 2021. **50**: p. 101395.
12. Ladj, A., et al., *A knowledge-based Digital Shadow for machining industry in a Digital Twin perspective.* Journal of Manufacturing Systems, 2021. **58**: p. 168-179.
13. Zhu, Z., et al., *Digital Twin-driven machining process for thin-walled part manufacturing.* Journal of Manufacturing Systems, 2021. **59**: p. 453-466.
14. Zhou, Y., et al., *Digital-twin-driven geometric optimization of centrifugal impeller with free-form blades for five-axis flank milling.* Journal of Manufacturing Systems, 2021. **58**: p. 22-35.
15. Cao, X., G. Zhao, and W. Xiao, *Digital Twin–oriented real-time cutting simulation for intelligent computer numerical control machining.* Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, 2022. **236**(1-2): p. 5-15.
16. Viola, J. and Y. Chen. *Digital Twin Enabled Smart Control Engineering as an Industrial AI: A New Framework and Case Study*. in *2020 2nd International Conference on Industrial Artificial Intelligence (IAI)*. 2020.
17. Wang, C.-P., et al., *In-process digital twin estimation for high-performance machine tools with coupled multibody dynamics.* CIRP Annals, 2020. **69**(1): p. 321-324.
18. Wu, L., J. Leng, and B. Ju, *Digital Twins-Based Smart Design and Control of Ultra-Precision Machining: A Review.* Symmetry, 2021. **13**(9): p. 1717.
19. Yang, X., et al., *A digital twin-driven hybrid approach for the prediction of performance degradation in transmission unit of CNC machine tool.* Robotics and Computer-Integrated Manufacturing, 2022. **73**: p. 102230.
20. Atalay, M., et al., *Digital twins in manufacturing: systematic literature review for physical–digital layer categorization and future research directions.* International Journal of Computer Integrated Manufacturing, 2022. **35**(7): p. 679-705.
21. Wilhelm, J., et al., *Review of Digital Twin-based Interaction in Smart Manufacturing: Enabling Cyber-Physical Systems for Human-Machine Interaction.* International Journal of Computer Integrated Manufacturing, 2021. **34**(10): p. 1031-1048.
22. Cimino, C., E. Negri, and L. Fumagalli, *Review of digital twin applications in manufacturing.* Computers in Industry, 2019. **113**: p. 103130.
23. Liu, M., et al., *Review of digital twin about concepts, technologies, and industrial applications.* Journal of Manufacturing Systems, 2021. **58**: p. 346-361.
24. Qiao, Q., et al., *Digital Twin for Machining Tool Condition Prediction.* Procedia CIRP, 2019. **81**: p. 1388-1393.

25. Christiand and G. Kiswanto, *Digital Twin Approach for Tool Wear Monitoring of Micro-Milling.* Procedia CIRP, 2020. **93**: p. 1532-1537.
26. Zhuang, K., et al., *Digital Twin-Driven Tool Wear Monitoring and Predicting Method for the Turning Process.* Symmetry, 2021. **13**(8): p. 1438.
27. Luo, W., et al., *Digital twin for CNC machine tool: modeling and using strategy.* Journal of Ambient Intelligence and Humanized Computing, 2019. **10**(3): p. 1129-1140.
28. Luo, W., et al. *Digital twin modeling method for CNC machine tool*. in *2018 IEEE 15th International Conference on Networking, Sensing and Control (ICNSC)*. 2018.
29. Ward, R., et al., *Machining Digital Twin using real-time model-based simulations and lookahead function for closed loop machining control.* The International Journal of Advanced Manufacturing Technology, 2021. **117**(11): p. 3615-3629.
30. Nguyen, T.N., S. Zeadally, and A. Vuduthala, *Cyber-Physical Cloud Manufacturing Systems with Digital-Twins.* IEEE Internet Computing, 2021: p. 1-1.
31. Malik, A.A. and A. Brem, *Digital twins for collaborative robots: A case study in human-robot interaction.* Robotics and Computer-Integrated Manufacturing, 2021. **68**: p. 102092.
32. Zheng, Y., S. Yang, and H. Cheng, *An application framework of digital twin and its case study.* Journal of Ambient Intelligence and Humanized Computing, 2019. **10**(3): p. 1141-1153.
33. Ghosh, A.K., et al., *Developing sensor signal-based digital twins for intelligent machine tools.* Journal of Industrial Information Integration, 2021. **24**: p. 100242.
34. Reisch, R.T., et al., *Context awareness in process monitoring of additive manufacturing using a digital twin.* The International Journal of Advanced Manufacturing Technology, 2022. **119**(5): p. 3483-3500.
35. Reisch, R.T., et al., *Context awareness in process monitoring of additive manufacturing using a digital twin.* 2022.
36. Rasheed, A., O. San, and T. Kvamsdal, *Digital Twin: Values, Challenges and Enablers From a Modeling Perspective.* IEEE Access, 2020. **8**: p. 21980-22012.
37. Jiang, Y., et al., *Industrial applications of digital twins.* Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 2021. **379**: p. 20200360.
38. Tao, F. and M. Zhang, *Digital Twin Shop-Floor: A New Shop-Floor Paradigm Towards Smart Manufacturing.* IEEE Access, 2017. **5**: p. 20418-20427.
39. Zhao, R., et al., *Digital Twin-Driven Cyber-Physical System for Autonomously Controlling of Micro Punching System.* IEEE Access, 2019. **7**: p. 9459-9469.
40. Botkina, D. and M. Xu, *ISO ToolMaker application development*, in *Degree Project in Production Engineering Management, Second Level*. 2017. p. 44.
41. Xu, J. and T. Guo, *Application and research on digital twin in electronic cam servo motion control system.* The International Journal of Advanced Manufacturing Technology, 2021. **112**(3): p. 1145-1158.
42. Holland, J.H., *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. 1992: The MIT Press.
43. He, B., T. Li, and J. Xiao, *Digital Twin-Driven Controller Tuning Method for Dynamics.* Journal of Computing and Information Science in Engineering, 2021. **21**(3).
44. Wang, G., Y. Cao, and Y. Zhang, *Digital twin-driven clamping force control for thin-walled parts.* Advanced Engineering Informatics, 2022. **51**: p. 101468.
45. Botkina, D., et al., *Digital Twin of a Cutting Tool.* Procedia CIRP, 2018. **72**: p. 215-218.
46. Luo, W., et al., *A hybrid predictive maintenance approach for CNC machine tool driven by Digital Twin.* Robotics and Computer-Integrated Manufacturing, 2020. **65**: p. 101974.
47. Hänel, A., et al., *Digital Twins for High-Tech Machining Applications—A Model-Based Analytics-Ready Approach.* Journal of Manufacturing and Materials Processing, 2021. **5**(3): p. 80.
48. Guo, M., et al., *Design and research of digital twin machine tool simulation and monitoring system.* The International Journal of Advanced Manufacturing Technology, 2023. **124**(11): p. 4253-4268.
49. Zhang, H., Q. Qi, and F. Tao, *A multi-scale modeling method for digital twin shop-floor.* Journal of Manufacturing Systems, 2022. **62**: p. 417-428.

50. Gunes, V., et al., *A Survey on Concepts, Applications, and Challenges in Cyber-Physical Systems.* KSII Transactions on Internet and Information Systems, 2014. **8**: p. 4242-4268.
51. Juarez, M.G., V.J. Botti, and A.S. Giret, *Digital Twins: Review and Challenges.* Journal of Computing and Information Science in Engineering, 2021. **21**(3).
52. Liang, Z., et al., *The process correlation interaction construction of Digital Twin for dynamic characteristics of machine tool structures with multi-dimensional variables.* Journal of Manufacturing Systems, 2022. **63**: p. 78-94.
53. Xin Tong, Q.L., Shiwei Pi, Yao Xiao, *Real-time machining data application and service based on IMT digital twin.* 2019.
54. Chi Ma, H.G., Jialan Liu, *Self learning-empowered thermal error control method of precision machine tools based on digital twin.* 2021.
55. Parris, C. *Everything you need to know about Digital Twins*. [cited 2022 01/02/22]; Available from: https://www.ge.com/digital/blog/what-digital-twin.
56. Lyu, D., et al., *Dynamic error of CNC machine tools: a state-of-the-art review.* The International Journal of Advanced Manufacturing Technology, 2020. **106**(5): p. 1869-1891.
57. Li, Y., et al., *A review on spindle thermal error compensation in machine tools.* International Journal of Machine Tools and Manufacture, 2015. **95**: p. 20-38.
58. Wang, S.-M., H.-J. Yu, and H.-W. Liao, *A new high-efficiency error compensation system for CNC multi-axis machine tools.* The International Journal of Advanced Manufacturing Technology, 2006. **28**(5): p. 518-526.
59. Zhao, W., *New evaluation method on the precision of NC machine tools.* 2018: p. Eng Sci 15(1):93–98.
60. Kiridena, V.S.B. and P.M. Ferreira, *Parameter estimation and model verification of first order quasistatic error model for three-axis machining centers.* International Journal of Machine Tools and Manufacture, 1994. **34**(1): p. 101-125.
61. Schmitz, T.L., et al., *Case study: A comparison of error sources in high-speed milling.* Precision Engineering, 2008. **32**(2): p. 126-133.
62. Schwenke, H., et al., *Geometric error measurement and compensation of machines—An update.* CIRP Annals, 2008. **57**(2): p. 660-675.
63. Ma, J., D. Lu, and W. Zhao, *Assembly errors analysis of linear axis of CNC machine tool considering component deformation.* The International Journal of Advanced Manufacturing Technology, 2016. **86**(1): p. 281-289.
64. Zhou, Z.-D., et al., *Actualities and Development of Heavy-Duty CNC Machine Tool Thermal Error Monitoring Technology.* Chinese Journal of Mechanical Engineering, 2017. **30**(5): p. 1262-1281.
65. Ramesh, R., M.A. Mannan, and A.N. Poo, *Error compensation in machine tools — a review: Part II: thermal errors.* International Journal of Machine Tools and Manufacture, 2000. **40**(9): p. 1257-1284.
66. Wang, S.-M., Y.-L. Liu, and Y. Kang, *An efficient error compensation system for CNC multi-axis machines.* International Journal of Machine Tools and Manufacture, 2002. **42**(11): p. 1235-1245.
67. *Total Guide to CNC Jigs, Fixtures, and Workholding Solutions for Mills*. 2020; Available from: https://www.cnccookbook.com/cnc-jigs-fixtures-workholding-solutions-milling/.
68. Liu, C., et al., *Dynamic and static error identification and separation method for three-axis CNC machine tools based on feature workpiece cutting.* The International Journal of Advanced Manufacturing Technology, 2020. **107**(5): p. 2227-2238.
69. FERREIRAt, V.S.B.K.a.P.M., *KINEMATIC MODELING OF QUASISTATIC ERRORS OF THREE-AXIS MACHINING CENTERS.* 1991.
70. Barre, P.-J., et al., *Influence of a Jerk Controlled Movement Law on the Vibratory Behaviour of High-Dynamics Systems.* Journal of Intelligent and Robotic Systems, 2005. **42**(3): p. 275-293.
71. Schneider, F., et al., *Sustainability in Ultra Precision and Micro Machining: A Review.* International Journal of Precision Engineering and Manufacturing-Green Technology, 2019. **6**(3): p. 601-610.

72. Zhang, S., et al., *Advances in ultra-precision machining of micro-structured functional surfaces and their typical applications.* International Journal of Machine Tools and Manufacture, 2019. **142**: p. 16-41.
73. Ullah, N., et al., *A comprehensive review of micro-milling: fundamental mechanics, challenges, and future prospective.* The International Journal of Advanced Manufacturing Technology, 2025. **137**(9): p. 4309-4351.
74. Kao, Y.C., et al. *Application of a cyber-physical system and machine-to-machine communication for metal processes*. in *2018 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*. 2018.
75. Xu, Z., et al., *Digital-twin-driven intelligent tracking error compensation of ultra-precision machining.* Mechanical Systems and Signal Processing, 2024. **219**: p. 111630.
76. Zhang, C., et al., *A digital twin defined autonomous milling process towards the online optimal control of milling deformation for thin-walled parts.* The International Journal of Advanced Manufacturing Technology, 2023. **124**(7): p. 2847-2861.
77. Tong, X., et al., *A digital twin-driven cutting force adaptive control approach for milling process.* Journal of Intelligent Manufacturing, 2025. **36**(1): p. 551-568.
78. Xu, M., K. Nakamoto, and Y. Takeuchi, *An Efficient Ultraprecision Machining System Automating Setting Operations of Roughly Machined Workpiece.* Journal of Manufacturing and Materials Processing, 2021. **5**(1): p. 11.
79. Hauptfleischová, B., et al., *In-Process Chatter Detection in Milling: Comparison of the Robustness of Selected Entropy Methods.* Journal of Manufacturing and Materials Processing, 2022. **6**(5): p. 125.
80. Chen, D., et al., *Development of a novel online chatter monitoring system for flexible milling process.* Mechanical Systems and Signal Processing, 2021. **159**: p. 107799.
81. Cao, H., K. Zhou, and X. Chen, *Chatter identification in end milling process based on EEMD and nonlinear dimensionless indicators.* International Journal of Machine Tools and Manufacture, 2015. **92**: p. 52-59.
82. Xia, G., et al., *Vertebral Lamina State Estimation in Robotic Bone Milling Process via Vibration Signals Fusion.* IEEE Transactions on Instrumentation and Measurement, 2022. **71**: p. 1-11.
83. Mili Hota, V.D.D., *Fabrication of A Test Rig for Experimental Studies on Misalignment Effect Between Rotors Connected Through Flexible Coupling by Torque and Vibration Measurements.* Stochastic Modeling & Applications Vol. 26 No. 3, 2022.
84. Mun, C.H., et al., *Indirect measurement of cutting forces during robotic milling using multiple sensors and a machine learning-based system identifier.* Journal of Manufacturing Processes, 2023. **85**: p. 963-976.
85. Dai, Y., et al., *Vibration Feedback Control for Robotic Bone Milling.* IEEE Transactions on Industrial Electronics, 2023. **70**(10): p. 10312-10322.
86. Lamraoui, M., *Spindle bearing fault detection in high-speed milling machines in non-stationary conditions.* Springer-Verlag, 2022.
87. Kumar, G.G., S.K. Sahoo, and P.K. Meher, *50 Years of FFT Algorithms and Applications.* Circuits, Systems, and Signal Processing, 2019. **38**(12): p. 5665-5698.
88. Scott, R., *Untangling the requirements of a Digital Twin*. 2020, Advanced Manufacturing Research Centre: Advanced Manufacturing Research Centre.
89. *qc20-w technical brochure*, Renishaw, Editor. 2013.
90. Ziegert, J.C. and C.D. Mize, *The laser ball bar: a new instrument for machine tool metrology.* Precision Engineering, 1994. **16**(4): p. 259-267.
91. Minh, D., et al., *Explainable artificial intelligence: a comprehensive review.* Artificial Intelligence Review, 2022. **55**(5): p. 3503-3568.
92. Deeks, A., *THE JUDICIAL DEMAND FOR EXPLAINABLE ARTIFICIAL INTELLIGENCE.* Columbia Law Review, 2019. **119**(7): p. 1829-1850.
93. Madathil, A.P., et al., *XAI-driven digital twin for cobot dynamic error compensation.* Procedia CIRP, 2024. **126**: p. 176-181.

94. Charlie Walker, X.L., Abhilash P M, Qi Liu, Rajeshkumar Madarkar, Erfu Yang, *Digital twin of dynamic error of a collaborative robot*, in *euspen's 23rd International Conference & Exhibition*. 2023: Copenhagen, Denmark.
95. Brolos, K. *What is the QLattice* 2024; Available from: https://docs.abzu.ai/docs/guides/getting_started/qlattice.
96. Wilstrup, C. and C. Cave, *Combining symbolic regression with the Cox proportional hazards model improves prediction of heart failure deaths.* BMC Med Inform Decis Mak, 2022. **22**(1): p. 196.
97. Broløs, K., et al., *An Approach to Symbolic Regression Using Feyn*. 2021.
98. Walker, C., *Digital Twin-Driven Work Handling*, in *ICPE2024: The 20thInternational Conference on Precision Engineering*. 2024: Sendia, Japan.
99. Piezotronics, P. *PCB Model 356B18*. [cited 2021; Available from: https://www.pcb.com/products?model=356b18.
100. *cDAQ-9174 Specifications.* Nathinal Instraments.
101. Renishaw. *Ballbar testing explained*. [cited 2025 02/01/2025]; information on cuducting ball bar tests]. Available from: https://www.renishaw.com/en/ballbar-testing-explained--6818.
102. Tony L. Schmitz a, John C. Ziegert b, J. Suzanne Canning c, Raul Zapata a, *Case study: A comparison of error sources in high-speed milling.* 2008.
103. John C. Ziegert, C.D.M., *The laser ball bar: a new instrument for machine tool metrology.* PRECISION ENGINEERING VOL 16 NO 4, 1994.
104. Renishaw. *QC20 ballbar for machine tool verification*. 2025 02/01/2025]; QC20 ballbar for machine tool verification with CNC machines]. Available from: https://www.renishaw.com/en/--11075
105. Usop, Z., et al., *Measuring of positioning, circularity and static errors of a CNC Vertical Machining Centre for validating the machining accuracy.* Measurement, 2015. **61**: p. 39-50.
106. Renishaw, *QC20 ballbar volumetric testing*
using a single program. 2022.

11 Appendix I Specification of Hardware in Dynamic Error Measurement

11.1.1 Accelerometers to be used

The accelerometers selected for the experiments involving the robotic arm and milling machine are the PCB Piezotronics 356B18. This tri-axial Integrated Electronics Piezo-Electric (IEPE) accelerometer is a type of piezoelectric sensor equipped with built-in impedance conversion electronics. The specifications of this accelerometer are shown in Figure 11-1[99]:

Performance	ENGLISH	SI
Sensitivity(± 10 %)	1000 mV/g	102 mV/(m/s²)
Measurement Range	± 5 g pk	± 49 m/s² pk
Frequency Range(± 5 %)	0.5 to 3000 Hz	0.5 to 3000 Hz
Frequency Range(± 10 %)	0.3 to 5000 Hz	0.3 to 5000 Hz
Resonant Frequency	≥ 20 kHz	≥ 20 kHz
Phase Response(± 5 °)(at 70°F [21°C])	2 to 2000 Hz	2 to 2000 Hz
Broadband Resolution(1 to 10,000 Hz)	0.00005 g rms	0.0005 m/s² rms
Non-Linearity	≤ 1 %	≤ 1 %
Transverse Sensitivity	≤ 5 %	≤ 5 %

Figure 11-1 PCB 356B18 Specifications [99]

The 356B18 accelerometer was chosen for its tri-axial capability, enabling simultaneous measurement of dynamic errors along the X, Y, and Z axes with a single device. Additionally, as a piezoelectric sensor, it is unaffected by orientation, which simplifies both testing procedures and operational setups.

11.1.2 Data logger

Data collection was conducted using a National Instruments (NI) NI-9234 module for sound and vibration measurements, connected through an NI cDAQ-9174 chassis. The specifications of these two devices are provided below in Figure 11-2.

Input Characteristics	
Number of channels	4 analog input channels
ADC resolution	24 bits
Type of ADC	Delta-Sigma (with analog prefiltering)
Sampling mode	Simultaneous
Type of TEDS supported	IEEE 1451.4 TEDS Class I
Internal master timebase (f_M)	
Frequency	13.1072 MHz
Accuracy	±50 ppm maximum
Data rate range (f_s)	
Using internal master timebase	
Minimum	1.652 kS/s
Maximum	51.2 kS/s
Using external master timebase	
Minimum	0.391 kS/s
Maximum	52.734 kS/s
Data rates [1] (f_s)	$(f_M \div 256)/n$, n = 1, 2, ..., 31
Input coupling	AC/DC (software-selectable)
AC cutoff frequency	
-3 dB	0.5 Hz
-0.1 dB	4.6 Hz maximum

Figure 11-2 NI cDAQ 9234 Specifications [100]

12 Appendix II Specification of Renishaw QC20-W

12.1.1 Ball bar Test

Ball bars are measurement tools that provide error readings for both quasi-static and dynamic measurements. The accuracy of these devices can reach up to 0.1 µm, as seen with the Renishaw QC20-W [89].

Ball bars function by measuring the linear distance between two precision spheres located at each end. This measurement can be obtained using either a laser or a transducer. By knowing the orientation of the ball bar, accurate data can be collected to assess the performance of a CNC machine[101-103][104].

The bullet points below shows what errors can be measured using a ball bar: [105]

- The flatness of the table
- The straightness of the table
- The parallelism of the table and spindle
- The parallelism of the table and spindle at Y-axis direction
- The parallelism of the T-slot of table and spindle at X-axis direction
- The perpendicular of the X-axis and Y-axis direction
- The straightness on Z-axis direction of the spindle
- The perpendicular on the spindle and the table
- The perpendicular to the centre line of spindle and the surface of table
- The circular run-out of the clutch hole of the spindle
- The circular run-out of the outer diameter surface of the spindle
- The circular run-out of the edge surface of the spindle
- The backlash of the X-axis
- The backlash of the Y-axis

There are two common tests that are used which are:

360^{o} testing

This test provides the foundational data for the X and Y coordinates. It requires the tool head to move in a circular path around the base of the ball bar, once in a clockwise direction and once counter clockwise. Conducting the test with a larger circular path can improve measurement accuracy [106].

Figure 12-1 360 degree test [106]

220^0 partial arc testing

"This test is used to measure the accuracy of the Z-axis within the X or Y plane. It requires a more precise setup, as the positioning of the ball bar is crucial. Once configured, the tool head moves in an arc over the base in the X plane to 220^0 and then returns to the starting position. The ball bar is then rotated 90° around the Z-axis, after which the tool head moves in an arc over the base in the Y plane to 220^0 and back [89].

Figure 12-2 220 degree test [89]

13 Appendix III Configuration and Specification of Hybrid Mill

Its configuration is shown in Figure 13-1, and its movement specifications are summarized in Table 13-1.

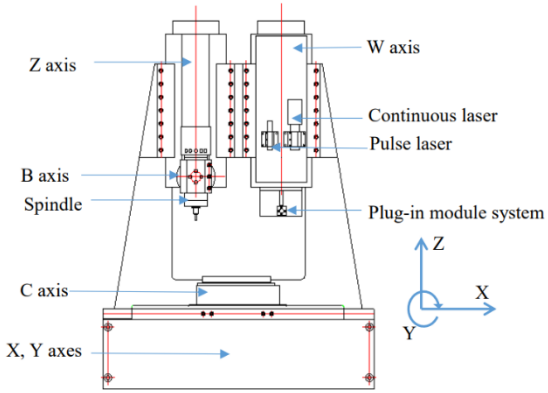


Figure 13-1 Hybrid mill axis layout

Table 13-1 Hybrid mill specifications

	X, Y, Z, W Axis			B Axis	C Axis	Spindle
Type	Air bearing slides			Ball bearing	Air bearing	Air Bearing
Stroke	X 400 (mm)	Y 150 (mm)	Z, W 150 (mm)	-10°~95°	360 °	N/A
Motion Accuracy	<0.1(μm)			20 (arcsec)	<10 (arcsec)	N/A
Accuracy Resolution	1 (nm)			2.4 (arcsec)	1 (arcsec)	N/A
Drive System	Brushless linear motor			Gear motor	Brushless Torque motor	Brushless motor
Maximum Speed	3000 (mm/min)			180 (deg/sec)	3000 (RPM)	10,000 (RPM)

14 Appendix IV Accelerometer Mounts for Hybrid mill

Figure 14-1 for the spindle, Figure 14-2 Hybrid mill table accelerometer mount for the table, and Figure 14-3 for the mill frame.

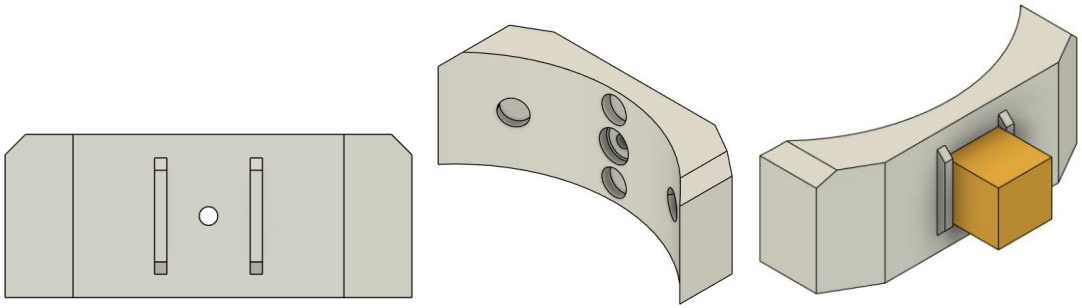


Figure 14-1 Hybrid mill spindle accelerometer mount

Figure 14-2 Hybrid mill table accelerometer mount

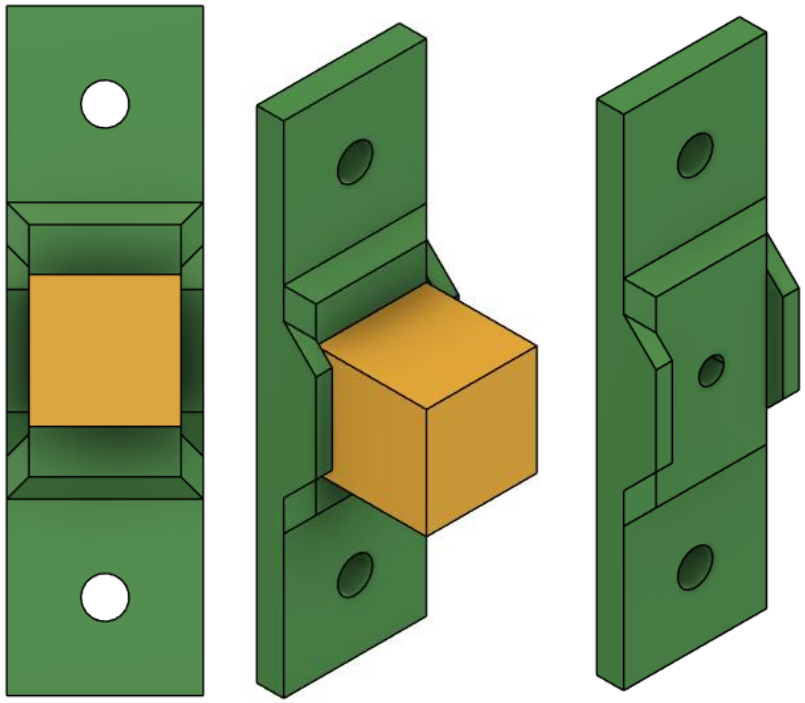


Figure 14-3 Hybrid mill frame accelerometer mount

15 Appendix V Cube Sat 3D technical drawing, printed part and internal structure

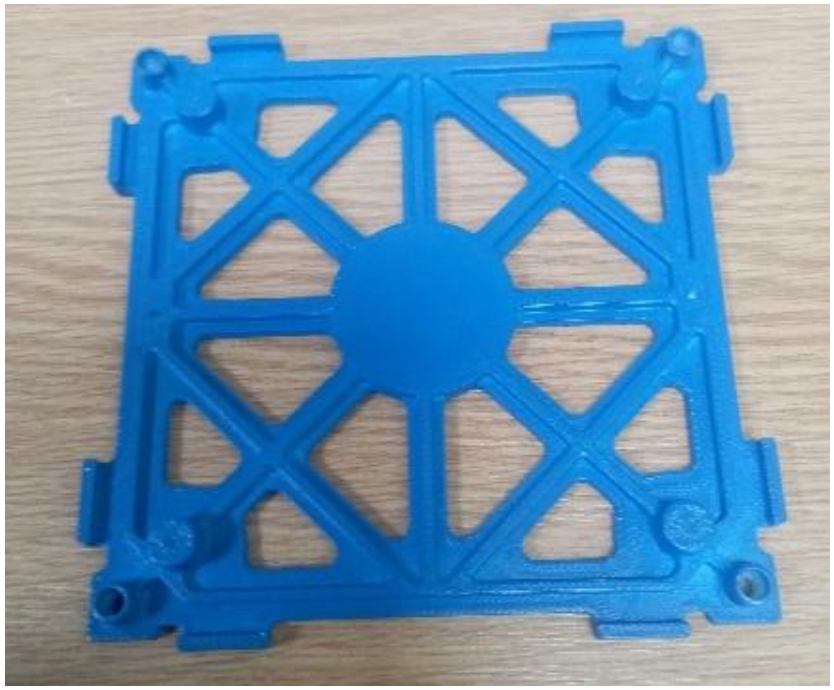


Figure 15-1 3D printed cube sat end cap

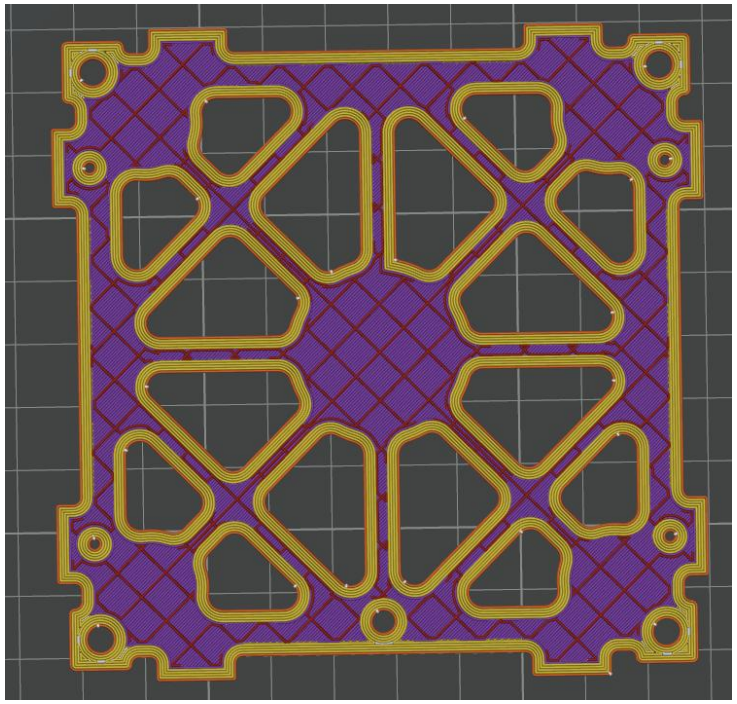


Figure 15-2 Cube Sat 3D printed structure

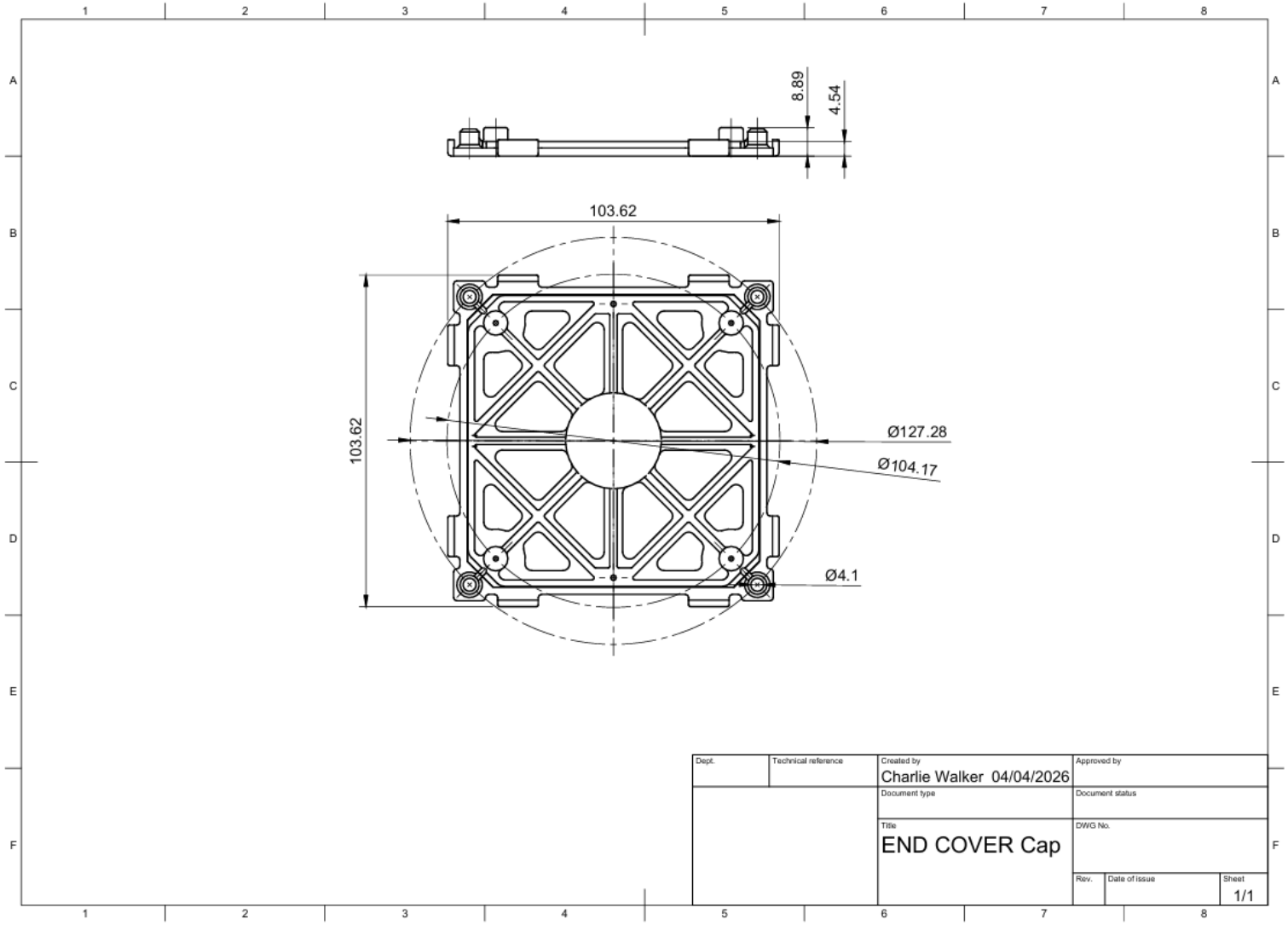


Figure 15-3 Cube Sat End Cap

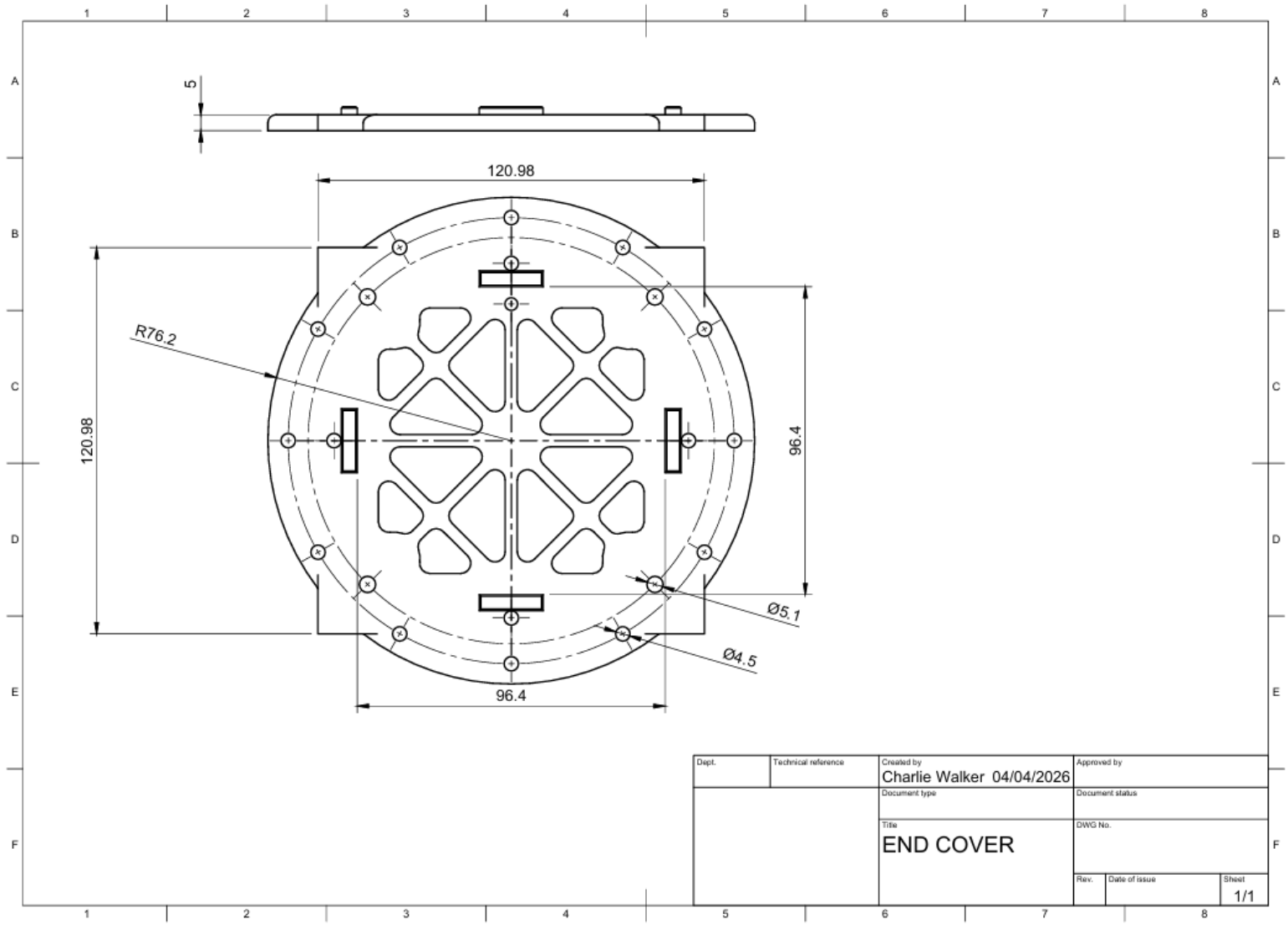


Figure 15-4 Cube Sat Technical Drawing Mount

16 Appendix VI Code

16.1 G-Code for running ball bar

```
N0010 G71 (input in mm)
N0020 G54 (set origin position)
N0030 G90 (absolute dimensions)
N0040 G17 (XY plane)
N0050 G64 (disable stopping between moves)
N0060 M05 (extra header code)
N0070 M19 (extra header code)
N0080 G94 F6000 (feedrate in mm/min)
N0090 G01 X-101.500 Y0.000 Z0.000 (move to start point)
N0100 M00 (stop to load ballbar)
N0110 G01 X-100.000 Y0.000 (in feed)
N0120 G03 X-100.000 Y0.000 I100.000 J0.000 (CCW arc)
N0130 G03 X-100.000 Y0.000 I100.000 J0.000 (CCW arc)
N0140 G01 X-101.500 Y0.000 (out feed)
N0150 G04 X5 (pause between runs)
N0160 G01 X-100.000 Y0.000 (in feed)
N0170 G02 X-100.000 Y0.000 I100.000 J0.000 (CW arc)
N0180 G02 X-100.000 Y0.000 I100.000 J0.000 (CW arc)
N0190 G01 X-101.500 Y0.000 (out feed)
N0200 M30 (end of program)
%
```

16.2 RoboDK Ball Bar code for COBOT

```
def BallBarTestXY():

  # Global parameters:

  global speed_ms    = 0.250

  global speed_rads  = 0.750

  global accel_mss   = 1.200

  global accel_radss = 1.200

  global blend_radius_m = 0.001

  global ref_frame = p[0,0,0,0,0,0]



  #-------------------------

  # Add any default subprograms here

  # For example, to drive a gripper as a program call:
```

```
# def Gripper_Open():
#   ...
# end
#
# Example to drive a spray gun:
def SprayOn(value):
  # use the value as an output:
  DO_SPRAY = 5
  if value == 0:
    set_standard_digital_out(DO_SPRAY, False)
  else:
    set_standard_digital_out(DO_SPRAY, True)
  end
```

```
end



# Example to drive an extruder:
def Extruder(value):
  # use the value as an output:
  if value < 0:
    # stop extruder
  else:
    # start extruder
  end
end



# Example to move an external axis
```

```
def MoveAxis(value):
 # use the value as an output:
 DO_AXIS_1 = 1
 DI_AXIS_1 = 1
 if value <= 0:
  set_standard_digital_out(DO_AXIS_1, False)


  # Wait for digital input to change state
  #while (get_standard_digital_in(DI_AXIS_1) != False):
  #  sync()
  #end
 else:
  set_standard_digital_out(DO_AXIS_1, True)
```

```
    # Wait for digital input to change state
    #while (get_standard_digital_in(DI_AXIS_1) != True):
    #  sync()
    #end
  end
end
#-------------------------



# Main program:
# Program generated by RoboDK v5.3.0 for UR10e on 16/01/2023 15:41:54
# Using nominal kinematics.
```

```
  ref_frame = p[-0.550000, 0.000000, 0.075000, -0.000000, 0.000000, 3.141593]
  set_tcp(p[0.000000, 0.000000, 0.158300, 0.000000, 0.000000, 0.000000])
  speed_ms    = 0.100
  sleep(0.500)
  blend_radius_m = 1.000
  movel(pose_trans(ref_frame,p[0.000000, 0.101500, 0.000000, 3.141593, 0.000000, -0.000000]),accel_mss,speed_ms,0,0)
  movel(pose_trans(ref_frame,p[0.000000, 0.100000, 0.000000, 3.141593, 0.000000, -0.000000]),accel_mss,speed_ms,0,0.001)
  movec(pose_trans(ref_frame,p[-0.100000, 0.000000, 0.000000, 3.141593, 0.000000, -0.000000]),pose_trans(ref_frame,p[0.000000, -0.100000,
0.000000, 3.141593, 0.000000, -0.000000]),accel_mss,speed_ms,0.028)
  # Skipping rounding for move with angle 180.0 deg
  movec(pose_trans(ref_frame,p[0.100000, 0.000000, 0.000000, 3.141593, 0.000000, -0.000000]),pose_trans(ref_frame,p[0.000000, 0.100000,
0.000000, 3.141593, 0.000000, -0.000000]),accel_mss,speed_ms,0)
  # Skipping rounding for move with angle 180.0 deg
```

movec(pose_trans(ref_frame,p[-0.100000, 0.000000, 0.000000, 3.141593, 0.000000, -0.000000]),pose_trans(ref_frame,p[0.000000, -0.100000, 0.000000, 3.141593, 0.000000, -0.000000]),accel_mss,speed_ms,0)

Skipping rounding for move with angle 180.0 deg

movec(pose_trans(ref_frame,p[0.100000, 0.000000, 0.000000, 3.141593, 0.000000, -0.000000]),pose_trans(ref_frame,p[0.000000, 0.100000, 0.000000, 3.141593, 0.000000, -0.000000]),accel_mss,speed_ms,0)

movel(pose_trans(ref_frame,p[0.000000, 0.101500, 0.000000, 3.141593, 0.000000, -0.000000]),accel_mss,speed_ms,0,0.001)

sleep(2.500)

Skipping rounding for move with angle 180.0 deg

movel(pose_trans(ref_frame,p[0.000000, 0.100000, 0.000000, 3.141593, 0.000000, -0.000000]),accel_mss,speed_ms,0,0)

movec(pose_trans(ref_frame,p[0.100000, 0.000000, 0.000000, 3.141593, 0.000000, -0.000000]),pose_trans(ref_frame,p[0.000000, -0.100000, 0.000000, 3.141593, 0.000000, -0.000000]),accel_mss,speed_ms,0.028)

Skipping rounding for move with angle 180.0 deg

movec(pose_trans(ref_frame,p[-0.100000, 0.000000, 0.000000, 3.141593, 0.000000, -0.000000]),pose_trans(ref_frame,p[0.000000, 0.100000, 0.000000, 3.141593, 0.000000, -0.000000]),accel_mss,speed_ms,0)

```
  # Skipping rounding for move with angle 180.0 deg
  movec(pose_trans(ref_frame,p[0.100000, 0.000000, 0.000000, 3.141593, 0.000000, -0.000000]),pose_trans(ref_frame,p[0.000000, -0.100000, 0.000000, 3.141593, 0.000000, -0.000000]),accel_mss,speed_ms,0)
  # Skipping rounding for move with angle 180.0 deg
  movec(pose_trans(ref_frame,p[-0.100000, 0.000000, 0.000000, 3.141593, 0.000000, -0.000000]),pose_trans(ref_frame,p[0.000000, 0.100000, 0.000000, 3.141593, 0.000000, -0.000000]),accel_mss,speed_ms,0)
  movel(pose_trans(ref_frame,p[0.000000, 0.101500, 0.000000, 3.141593, 0.000000, -0.000000]),accel_mss,speed_ms,0,0.001)
  # End of main program
end


BallBarTestXY()
```

16.3 ball bar measured results


```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<TEST_DOCUMENT>
 <TEST_SPEC NAME="XY 360deg 100mm Calibrated" version="2.0">
  <INSTRUMENT MANUFACTURER="Renishaw" MODEL="QC10">
   <DYNAMIC_BALLBAR CALIBRATION_REQUIRED="true">
    <NOMINAL_LENGTH>100</NOMINAL_LENGTH>
    <PROGRAMMED_FEEDRATE>5000</PROGRAMMED_FEEDRATE>
    <START_ANGLE>0</START_ANGLE>
    <END_ANGLE>360</END_ANGLE>
    <OVERSHOOT_ANGLE>180</OVERSHOOT_ANGLE>
    <TEST_RADIUS>100</TEST_RADIUS>
    <PLANE VIEW="+A+B" TEST_PLANE="XY">
     <A_AXIS_NAME>X</A_AXIS_NAME>
```

```
 <B_AXIS_NAME>Y</B_AXIS_NAME>
</PLANE>
<RUN_SPEC>
 <RUN_DIRECTION>CCW</RUN_DIRECTION>
 <RUN_DIRECTION>CW</RUN_DIRECTION>
 <FEATURE_SPEC NAME="AF_OFFSET_A" DT="UT_LENGTH_MM">
  <TOLERANCE LEVEL="WARNING">
   <LOWER_LIMIT>-0.1</LOWER_LIMIT>
   <UPPER_LIMIT>0.1</UPPER_LIMIT>
  </TOLERANCE>
 </FEATURE_SPEC>
 <FEATURE_SPEC NAME="AF_OFFSET_B" DT="UT_LENGTH_MM">
  <TOLERANCE LEVEL="WARNING">
   <LOWER_LIMIT>-0.1</LOWER_LIMIT>
   <UPPER_LIMIT>0.1</UPPER_LIMIT>
  </TOLERANCE>
```

```
 </FEATURE_SPEC>
 <FEATURE_SPEC NAME="AF_FEEDRATE" DT="UT_FEEDRATE">
  <TOLERANCE LEVEL="WARNING">
   <LOWER_LIMIT>4500</LOWER_LIMIT>
   <UPPER_LIMIT>5500</UPPER_LIMIT>
  </TOLERANCE>
 </FEATURE_SPEC>
 <FEATURE_SPEC NAME="AF_BEST_RADIUS" DT="UT_LENGTH_MM">
  <TOLERANCE LEVEL="WARNING">
   <LOWER_LIMIT>99.95</LOWER_LIMIT>
   <UPPER_LIMIT>100.05</UPPER_LIMIT>
  </TOLERANCE>
 </FEATURE_SPEC>
</RUN_SPEC>
<CENTRE></CENTRE>
<SPINDLE_NUMBER>1</SPINDLE_NUMBER>
```

```
    <SAMPLE_RATE>198.943678864869</SAMPLE_RATE>
    <AVERAGING ENABLED="True" />
    <NC_PROGRAM_ID></NC_PROGRAM_ID>
    <TEMP_MATERIAL_COEFFICIENT>11.7</TEMP_MATERIAL_COEFFICIENT>
   </DYNAMIC_BALLBAR>
  </INSTRUMENT>
  <ANALYSIS_SPEC NAME="RENISHAW_DIAGNOSTICS" />
  <ANALYSIS_SPEC NAME="ANSI_B_5_54" />
  <ANALYSIS_SPEC NAME="ANSI_B_5_57" />
  <ANALYSIS_SPEC NAME="ISO230_4" />
</TEST_SPEC>
<OPERATOR>kbb20178</OPERATOR>
<COMMENT NAME="PRE_TEST_COMMENT"></COMMENT>
<DATE_STARTED>20230214-110016</DATE_STARTED>
<DATE_COMPLETED>20230214-110111</DATE_COMPLETED>
<MACHINE_TESTED NAME="UR10e Error Map" />
```

<EQUIPMENT>

<BALLBAR>

<MANUFACTURER>Renishaw</MANUFACTURER>

<MODEL>QC20-W</MODEL>

<SERIAL_NUMBER>3RGK88</SERIAL_NUMBER>

<SCALE_FACTOR>1</SCALE_FACTOR>

<CERTIFICATE_NUMBER>3RGK88-211011-00</CERTIFICATE_NUMBER>

<LAST_CALIBRATED>20211011</LAST_CALIBRATED>

</BALLBAR>

<CALIBRATOR>

<MANUFACTURER>Renishaw</MANUFACTURER>

<MODEL>ZERODUR</MODEL>

<SERIAL_NUMBER>3W3C55</SERIAL_NUMBER>

<EXPANSION_COEFFICIENT>0.0</EXPANSION_COEFFICIENT>

<CALIBRATED_LENGTH>100.0015</CALIBRATED_LENGTH>

<TEMPERATURE>20.0</TEMPERATURE>

```
    <CERTIFICATE_NUMBER></CERTIFICATE_NUMBER>
    <LAST_CALIBRATED>20221014</LAST_CALIBRATED>
  </CALIBRATOR>
  <SOFTWARE>
    <AUTHOR>Renishaw</AUTHOR>
    <TITLE>Renishaw Ballbar 20</TITLE>
    <VERSION>5.09.09.03</VERSION>
  </SOFTWARE>
</EQUIPMENT>
<RUN_RESULTS>
  <BALLBAR_RUN>
    <RUN_DIRECTION>CCW</RUN_DIRECTION>
    <RUN_TEMPERATURE>20.0</RUN_TEMPERATURE>
    <START_INDEX>796</START_INDEX>
    <END_INDEX>2385</END_INDEX>
    <SAMPLE_RATE>200.000</SAMPLE_RATE>
```

<READINGS>-0.24842 -0.29488 -0.29135 -0.24568 -0.15190 -0.04369 +0.07413 +0.20343 +0.31911 +0.41617 +0.49736 +0.55387 +0.57042 +0.54823 +0.49386 +0.41280 +0.31745 +0.21866 +0.12299 +0.03318 -0.04259 -0.09549 -0.11869 -0.10776 -0.06055 +0.01111 +0.09413 +0.18018 +0.25954 +0.32566 +0.37232 +0.39817 +0.40160 +0.37857 +0.33013 +0.26560 +0.18195 +0.09553 +0.01686 -0.05304 -0.10575 -0.14152 -0.16191 -0.16960 -0.16276 -0.13926 -0.10159 -0.05304 -0.00062 +0.04992 +0.10085 +0.14299 +0.16837 +0.17675 +0.16648 +0.14299 +0.11314 +0.07964 +0.04351 +0.00592 -0.02989 -0.05823 -0.07387 -0.07943 -0.07827 -0.07063 -0.05206 -0.02304 +0.01399 +0.05512 +0.09443 +0.13002 +0.16164 +0.18887 +0.21010 +0.22001 +0.21720 +0.20374 +0.18514 +0.16348 +0.13907 +0.11333 +0.08691 +0.06405 +0.04779 +0.03746 +0.03477 +0.03776 +0.04314 +0.05188 +0.06588 +0.08190 +0.09602 +0.11064 +0.12440 +0.13363 +0.13962 +0.14042 +0.13553 +0.12807 +0.11700 +0.10434 +0.09266 +0.08061 +0.07120 +0.06710 +0.06845 +0.07444 +0.08496 +0.09957 +0.11498 +0.12978 +0.14232 +0.15351 +0.16177 +0.16605 +0.16678 +0.16403 +0.15730 +0.14531 +0.12892 +0.11272 +0.09786 +0.08428 +0.07053 +0.05805 +0.04791 +0.03801 +0.03165 +0.03147 +0.03281 +0.03391 +0.03672 +0.04106 +0.04620 +0.05072 +0.05628 +0.06276 +0.07004 +0.07902 +0.09033 +0.10152 +0.11174 +0.12048 +0.12794 +0.13186 +0.13412 +0.13143 +0.12384 +0.11589 +0.10862 +0.10189 +0.09559 +0.09046 +0.08526 +0.07902 +0.07254 +0.06612 +0.05958 +0.05359 +0.05176 +0.05432 +0.05909 +0.06227 +0.06056 +0.05750 +0.05316 +0.04626 +0.03788 +0.02780 +0.01411 -0.00129 -0.01584 -0.02543 -0.02738 -0.02671 -0.02360 -0.01840 -0.00954 +0.00415 +0.01936 +0.03654 +0.05671 +0.07701 +0.09162 +0.10201 +0.11241 +0.12195 +0.12941 +0.13283 +0.13265 +0.12702 +0.11791 +0.11131 +0.10299 +0.09370 +0.08483 +0.07352 +0.06227 +0.05359 +0.04595 +0.03886 +0.03446 +0.03360 +0.03923 +0.04797 +0.05708 +0.06777 +0.07945 +0.08929 +0.09529 +0.09596 +0.09070 +0.07658 +0.06032 +0.04424 +0.02816 +0.01673 +0.00653 -0.00050 -0.00252 -0.00374 -0.00496 -0.00331 +0.00127 +0.00659 +0.01741 +0.03586 +0.05763 +0.08037 +0.10189 +0.11571 +0.12702 +0.13852 +0.14672 +0.15014 +0.14965 +0.14721 +0.14158 +0.13296 +0.12421 +0.11167 +0.09810 +0.08459 +0.07254 +0.06288 +0.05426 +0.04674 +0.04063 +0.03855 +0.03892 +0.04002 +0.04130 +0.04540 +0.05286 +0.06117 +0.06814 +0.07315 +0.07529 +0.07401 +0.06490 +0.04931 +0.03214 +0.01240 -0.00600 -0.02115 -0.03062 -0.03362 -0.03313 -0.03013 -0.02384 -0.01681 -0.00698 +0.00427 +0.01588 +0.02761 +0.03764 +0.04736 +0.05518 +0.05665 +0.05683 +0.05665 +0.05506 +0.05176 +0.04858 +0.04619 +0.04656 +0.05274 +0.06087 +0.06997 +0.08025 +0.08942 +0.09889 +0.10819 +0.11675 +0.12495 +0.13094 +0.13412 +0.13657 +0.13601 +0.13149 +0.12280 +0.10843 +0.09113 +0.07199 +0.05231 +0.03415 +0.01979 +0.01044 +0.00586 +0.00610 +0.01038 +0.01606 +0.02113 +0.02902 +0.03910 +0.04870 +0.05897 +0.06924 +0.07774 +0.08116 +0.07951 +0.07633 +0.07334 +0.06942 +0.06533 +0.06337 +0.06294 +0.06484 +0.06820 +0.07107 +0.07401 +0.07535 +0.07456 +0.07218 +0.06765 +0.06178 +0.05231 +0.03935 +0.02382 +0.00616 -0.00887 -0.01901 -0.02574 -0.03099 -0.03227 -0.03087 -0.02842 -0.02476 -0.01975 -0.01388 -0.00997 -0.00789 -0.00722 -0.00887 -0.01205 -0.01498 -0.01822 -0.02164 -0.02610 -0.03032 -0.03233 -0.03197 -0.03087 -0.02818 -0.02085 -0.01407 -0.00606 +0.00402 +0.01154 +0.01881 +0.02566 +0.03232 +0.03886 +0.04442 +0.04907 +0.04931 +0.04821 +0.04723 +0.04473 +0.04350 +0.04289 +0.04338 +0.04442 +0.04509 +0.04558 +0.04399 +0.04136 +0.04112 +0.03996 +0.03739 +0.03452 +0.03232 +0.02883 +0.02504 +0.02181

+0.02009 +0.01887 +0.01771 +0.01759 +0.01649 +0.01624 +0.01539 +0.01441 +0.01337 +0.01080 +0.00653 -0.00044 -0.01211 -0.02677 -0.04553 -0.06728 -0.08737 -0.10550 -
0.12040 -0.13017 -0.13457 -0.13396 -0.13438 -0.12883 -0.12083 -0.11717 -0.11393 -0.11613 -0.11839 -0.12455 -0.13389 -0.14452 -0.15587 -0.16673 -0.17918 -0.19224 -0.20377 -
0.21250 -0.21896 -0.22281 -0.22384 -0.21896 -0.21079 -0.20127 -0.18950 -0.17729 -0.16338 -0.14745 -0.13231 -0.11924 -0.10691 -0.09543 -0.08389 -0.07070 -0.05579 -0.04248 -
0.03392 -0.02824 -0.02610 -0.02745 -0.02952 -0.03307 -0.03667 -0.03777 -0.03967 -0.03942 -0.03728 -0.03606 -0.03557 -0.03588 -0.03740 -0.04113 -0.04614 -0.05139 -0.05952 -
0.06728 -0.07259 -0.07821 -0.08346 -0.08780 -0.09152 -0.09360 -0.09317 -0.09073 -0.08621 -0.07888 -0.07070 -0.06532 -0.06196 -0.05946 -0.05787 -0.05677 -0.05646 -0.05610 -
0.05689 -0.05720 -0.05524 -0.05182 -0.04657 -0.04076 -0.03637 -0.03521 -0.03563 -0.03722 -0.03826 -0.03912 -0.03875 -0.03740 -0.03612 -0.03551 -0.03508 -0.03630 -0.04040 -
0.04687 -0.05384 -0.05988 -0.06202 -0.06007 -0.05543 -0.04956 -0.04309 -0.03857 -0.03514 -0.03172 -0.03044 -0.02952 -0.02940 -0.03032 -0.03081 -0.03130 -0.03356 -0.03808 -
0.04394 -0.05029 -0.05494 -0.05958 -0.06367 -0.06709 -0.06886 -0.06746 -0.06422 -0.06105 -0.05940 -0.05934 -0.06245 -0.06880 -0.07735 -0.08713 -0.09574 -0.10227 -0.10581 -
0.10642 -0.10673 -0.10758 -0.10825 -0.11118 -0.11387 -0.11418 -0.11418 -0.11332 -0.10978 -0.10648 -0.10514 -0.10496 -0.10441 -0.10093 -0.09812 -0.09574 -0.09409 -0.09262 -
0.09061 -0.08639 -0.07906 -0.06966 -0.06092 -0.05537 -0.04944 -0.04486 -0.04358 -0.04523 -0.04999 -0.05616 -0.06215 -0.06569 -0.06630 -0.06343 -0.05934 -0.05708 -0.05640 -
0.05634 -0.05708 -0.05970 -0.06416 -0.06789 -0.06984 -0.06996 -0.06948 -0.06929 -0.06948 -0.07100 -0.07332 -0.07387 -0.07442 -0.07442 -0.07326 -0.07088 -0.06819 -0.06422 -
0.06080 -0.06050 -0.06202 -0.06447 -0.06569 -0.06544 -0.06331 -0.06025 -0.05756 -0.05775 -0.06001 -0.06117 -0.06080 -0.05866 -0.05311 -0.04510 -0.03808 -0.03527 -0.03563 -
0.03771 -0.03942 -0.03918 -0.03741 -0.03686 -0.03961 -0.04706 -0.05799 -0.07082 -0.08297 -0.09097 -0.09629 -0.09812 -0.09781 -0.09720 -0.09732 -0.09665 -0.09415 -0.09079 -
0.08499 -0.07778 -0.07381 -0.07326 -0.07467 -0.07851 -0.08303 -0.08621 -0.08902 -0.09372 -0.09861 -0.10300 -0.10496 -0.10459 -0.10196 -0.09848 -0.09659 -0.09500 -0.09378 -
0.09183 -0.09024 -0.08664 -0.08077 -0.07516 -0.07039 -0.06813 -0.06923 -0.07247 -0.07540 -0.07803 -0.07998 -0.08126 -0.08255 -0.08456 -0.08633 -0.08658 -0.08462 -0.08285 -
0.08206 -0.08138 -0.08145 -0.08236 -0.08242 -0.08187 -0.08084 -0.08004 -0.08084 -0.08395 -0.08761 -0.09146 -0.09586 -0.10019 -0.10386 -0.10752 -0.11247 -0.12089 -0.13200 -
0.14391 -0.15557 -0.16631 -0.17442 -0.17979 -0.18474 -0.19005 -0.19481 -0.19792 -0.19865 -0.19651 -0.19188 -0.18516 -0.17876 -0.17217 -0.16454 -0.15605 -0.14629 -0.13493 -
0.12205 -0.10819 -0.09500 -0.08578 -0.07894 -0.07192 -0.06434 -0.05512 -0.04687 -0.04309 -0.04498 -0.04865 -0.05249 -0.05683 -0.06263 -0.07009 -0.07674 -0.08267 -0.08658 -
0.08761 -0.08456 -0.08059 -0.07589 -0.06838 -0.06019 -0.05024 -0.04071 -0.03350 -0.02775 -0.02164 -0.01608 -0.01364 -0.01474 -0.01993 -0.02708 -0.03338 -0.03826 -0.04358 -
0.04981 -0.05476 -0.05824 -0.06184 -0.06508 -0.06502 -0.06374 -0.06141 -0.05824 -0.05512 -0.05134 -0.04828 -0.04639 -0.04419 -0.04211 -0.04297 -0.04419 -0.04510 -0.04853 -
0.05231 -0.05549 -0.05897 -0.06190 -0.06312 -0.06178 -0.05805 -0.05610 -0.05653 -0.05531 -0.05158 -0.04846 -0.04468 -0.04260 -0.04321 -0.04425 -0.04364 -0.04132 -0.04071 -
0.02501 -0.03857 -0.18962 -0.16918 -0.02054 -0.08719 -0.16283 -0.12395 -0.13023 -0.13213 -0.10349 -0.12926 -0.15728 -0.11741 -0.04382 +0.02254 +0.05359 +0.05267 +0.02217

-0.02549 -0.05121 -0.05219 -0.04908 -0.05305 -0.08597 -0.14647 -0.20518 -0.23513 -0.22013 -0.17925 -0.12987 -0.07803 -0.03447 +0.00011 +0.01759 +0.01997 +0.02235 +0.00622 -0.01987 -0.04254 -0.07784 -0.12877 -0.18816 -0.23903 -0.26148 -0.25337 -0.23049 -0.20786 -0.19029 -0.17925 -0.17058 -0.13957 -0.08780 -0.04205 -0.00576 +0.02406 +0.04595 +0.06288 +0.07315 +0.06869 +0.05071 +0.02608 -0.00435 -0.03564 -0.06465 -0.09336 -0.11723 -0.13127 -0.13249 -0.12285 -0.11039 -0.10129 -0.09855 -0.09757 -0.08914 -0.07161 -0.05103 -0.03063 -0.01486 -0.00521 -0.00056 +0.00249 +0.00720 +0.01117 +0.00860 +0.00225 -0.00594 -0.01578 -0.02409 -0.03026 -0.03820 -0.04853 -0.05457 -0.05732 -0.05873 -0.06031 -0.06355 -0.06410 -0.05799 -0.04914 -0.04144 -0.03643 -0.03454 -0.03368 -0.03313 -0.03527 -0.03637 -0.03619 -0.04064 -0.05005 -0.05891 -0.06764 -0.07583 -0.08016 -0.08181 -0.08181 -0.07980 -0.07742 -0.07027 -0.05708 -0.04468 -0.03417 -0.02397 -0.01578 -0.01065 -0.00661 -0.00307 +0.00047 +0.00341 -0.00099 -0.01498 -0.03276 -0.05024 -0.06819 -0.08676 -0.10471 -0.11882 -0.12602 -0.12505 -0.12065 -0.11253 -0.10001 -0.08908 -0.08120 -0.07357 -0.06685 -0.06325 -0.06038 -0.05940 -0.06172 -0.06306 -0.06484 -0.07168 -0.07839 -0.08200 -0.08847 -0.09342 -0.09464 -0.09592 -0.09268 -0.08585 -0.07907 -0.07162 -0.06300 -0.05244 -0.03985 -0.03154 -0.02696 -0.02183 -0.01877 -0.01951 -0.02440 -0.03020 -0.03570 -0.04321 -0.05409 -0.06594 -0.07962 -0.09116 -0.09910 -0.10239 -0.10148 -0.09751 -0.08920 -0.07931 -0.06911 -0.06087 -0.05158 -0.04022 -0.03142 -0.02757 -0.02580 -0.02696 -0.03075 -0.03215 -0.03264 -0.03496 -0.03784 -0.04187 -0.04480 -0.04700 -0.04895 -0.05036 -0.05115 -0.04956 -0.04743 -0.04743 -0.04480 -0.03490 -0.01902 -0.00270 +0.01202 +0.02461 +0.03256 +0.03262 +0.03201 +0.03207 +0.02541 +0.01575 +0.00530 -0.00875 -0.02470 -0.03716 -0.04419 -0.04559 -0.04486 -0.04248 -0.03918 -0.03973 -0.04223 -0.04694 -0.05360 -0.05757 -0.06093 -0.06300 -0.06221 -0.06148 -0.06025 -0.05622 -0.05201 -0.04840 -0.04541 -0.04309 -0.04126 -0.04101 -0.04095 -0.04346 -0.04902 -0.05360 -0.05622 -0.05653 -0.05592 -0.05518 -0.05531 -0.05457 -0.05372 -0.05054 -0.04474 -0.04034 -0.03564 -0.03185 -0.02776 -0.02488 -0.02531 -0.02391 -0.01920 -0.01297 -0.00619 +0.00121 +0.00726 +0.01172 +0.01746 +0.02327 +0.02541 +0.02620 +0.02791 +0.02846 +0.02852 +0.02828 +0.02871 +0.03005 +0.03250 +0.03403 +0.03201 +0.02632 +0.02009 +0.01263 +0.00438 -0.00148 -0.00716 -0.01175 -0.01676 -0.02171 -0.02708 -0.03179 -0.03600 -0.03814 -0.03796 -0.03619 -0.03289 -0.02574 -0.01627 -0.00802 -0.00099 +0.00518 +0.01050 +0.01435 +0.01728 +0.01716 +0.01508 +0.01337 +0.01160 +0.01184 +0.01251 +0.01300 +0.01367 +0.01325 +0.01160 +0.00726 +0.00206 -0.00277 -0.00784 -0.01132 -0.01254 -0.01242 -0.01120 -0.00784 -0.00484 -0.00130 +0.00414 +0.00891 +0.01361 +0.01978 +0.02345 +0.02345 +0.02321 +0.02223 +0.01948 +0.01520 +0.01080 +0.00487 -0.00124 -0.00490 -0.00521 -0.00411 +0.00139 +0.00970 +0.01948 +0.02962 +0.03867 +0.04704 +0.05316 +0.05634 +0.05903 +0.06055 +0.05994 +0.05676 +0.05059 +0.04405 +0.03690 +0.02822 +0.01868 +0.01007 +0.00200 -0.00484 -0.01095 -0.01584 -0.01768 -0.01780 -0.01755 -0.01523 -0.01254 -0.01034 -0.00741 -0.00484 -0.00222 +0.00078 +0.00145 +0.00157 +0.00176 +0.00188 +0.00341 +0.00451 +0.00634 +0.00823 +0.00854 +0.00915 +0.00909 +0.00903 +0.00884 +0.00897 +0.00915 +0.00982 +0.01147 +0.01123 +0.01080 +0.01153 +0.01196 +0.01263 +0.01386 +0.01551 +0.01563 +0.01404 +0.01251 +0.01001 +0.00499 -0.00325 -0.01358 -0.02782 -0.04474 -0.06349 -0.08023 -0.09452 -0.10679 -0.11436 -0.11986 -0.12248 -0.12206 -0.12181 -0.11992 -0.10960 -0.10392 -0.10233 -0.10093 -0.10130 -0.10447 -0.11455 -0.12132 -0.12651 -0.13188 -0.13744 -

0.13842 -0.13945 -0.14068 -0.13964 -0.13836 -0.13451 -0.12731 -0.11913 -0.10978 -0.10001 -0.08963 -0.07705 -0.06343 -0.04853 -0.03338 -0.01847 -0.00484 +0.00952 +0.02247 +0.03158 +0.03500 +0.03739 +0.03879 +0.03763 +0.03616 +0.03280 +0.02663 +0.01862 +0.00860 -0.00142 -0.00998 -0.01810 -0.02598 -0.03124 -0.03460 -0.03600 -0.03668 -0.03796 -0.03961 -0.04016 -0.04169 -0.04242 -0.04346 -0.04407 -0.04346 -0.04126 -0.03704 -0.03185 -0.02550 -0.01902 -0.01431 -0.00961 -0.00588 -0.00185 +0.00114 +0.00359 +0.00671 +0.00909 +0.01117 +0.01453 +0.01771 +0.02217 +0.02632 +0.02944 +0.03317 +0.03641 +0.03928 +0.04185 +0.04442 +0.04955 +0.05420 +0.05866 +0.06227 +0.06789 +0.07351 +0.07743 +0.07908 +0.07841 +0.07437 +0.06820 +0.06110 +0.05334 +0.04631 +0.04209 +0.04105 +0.04167 +0.04582 +0.04998 +0.05145 +0.05285 +0.05126 +0.04857 +0.04545 +0.04185 +0.03995 +0.03971 +0.04050 +0.04179 +0.04374 +0.04600 +0.04912 +0.05132 +0.05322 +0.05585 +0.05572 +0.05426 +0.05151 +0.04637 +0.04203 +0.03836 +0.03512 +0.03439 +0.03506 +0.03763 +0.04087 +0.04380 +0.04643 +0.04600 +0.04417 +0.04191 +0.03689 +0.02987 +0.02296 +0.01715 +0.01159 +0.00597 -0.00045 -0.00650 -0.01193 -0.01682 -0.02226 -0.02837 -0.03460 -0.04028 -0.04395 -0.04682 -0.04554 -0.04254 -0.03967 -0.03613 -0.03167 -0.02751 -0.02428 -0.02257 -0.01976 -0.01389 -0.00888 -0.00314 +0.00469 +0.01446 +0.02369 +0.02962 +0.03311 +0.03280 +0.03152 +0.02797 +0.02192 +0.01795 +0.01520 +0.01049 +0.00499 +0.00102 -0.00045 -0.00039 +0.00475 +0.01361 +0.02412 +0.03598 +0.04991 +0.06257 +0.07235 +0.07950 +0.08446 +0.08770 +0.08990 +0.09247 +0.09314 +0.09033 +0.08446 +0.07724 +0.06954 +0.06440 +0.06300 +0.06501 +0.06935 +0.07498 +0.07999 +0.08018 +0.07657 +0.07125 +0.06520 +0.06092 +0.05731 +0.05297 +0.04429 +0.03164 +0.01825 +0.00469 -0.00472 -0.00644 -0.00191 +0.00408 +0.01000 +0.01495 +0.01844 +0.01868 +0.01935 +0.02241 +0.02681 +0.03133 +0.03555 +0.03903 +0.04026 +0.03928 +0.03958 +0.04087 +0.04264 +0.04380 +0.04594 +0.04943 +0.05138 +0.05218 +0.05560 +0.06098 +0.06795 +0.07455 +0.07859 +0.07889 +0.07620 +0.07033 +0.06226 +0.05456 +0.05101 +0.05120 +0.04991 +0.04722 +0.04551 +0.04527 +0.04820 +0.05285 +0.05829 +0.06202 +0.06183 +0.05896 +0.05297 +0.04649 +0.04331 +0.04350 +0.04478 +0.04735 +0.04949 +0.04784 +0.04509 +0.04350 +0.04191 +0.04160 +0.04368 +0.04661 +0.05034 +0.05529 +0.06183 +0.06984 +0.07901 +0.08837 +0.09522 +0.09858 +0.09950 +0.09889 +0.09626 +0.09081 +0.08446 +0.07718 +0.06935 +0.05988 +0.05126 +0.04490 +0.03836 +0.03231 +0.02987 +0.03145 +0.03567 +0.04374 +0.05303 +0.06061 +0.06721 +0.07162 +0.07308 +0.07272 +0.07180 +0.06917 +0.06538 +0.06196 +0.05823 +0.05273 +0.04722 +0.04258 +0.03867 +0.03738 +0.03873 +0.04160 +0.04722 +0.05633 +0.06728 +0.07840 +0.09014 +0.10048 +0.10763 +0.11044 +0.10879 +0.10433 +0.09766 +0.08929 +0.08109 +0.07333 +0.06556 +0.05865 +0.05419 +0.05144 +0.05034 +0.05058 +0.05132 +0.05144 +0.05107 +0.04802 +0.04154 +0.03292 +0.02149 +0.00737 -0.00839 -0.02532 -0.04310 -0.06105 -0.07773 -0.09281 -0.10387 -0.10936 -0.11022 -0.10741 -0.10130 -0.09232 -0.08121 -0.06924 -0.05696 -0.04737 -0.03955 -0.03369 -0.02898 -0.02410 -0.01841 -0.01181 -0.00277 +0.00878 +0.02100 +0.03426 +0.04887 +0.06299 +0.07681 +0.08996 +0.10139 +0.11166 +0.11723 +0.11656 +0.11093 +0.11900 +0.12389 +0.10359 +0.18084 +0.24735 +0.26731 +0.28340 +0.27343 +0.22495 +0.15882 +0.11038 +0.10628 +0.11992 +0.12151 +0.12744 +0.14267 +0.14591 +0.13013 +0.11307 +0.11368 +0.13894 +0.17680 +0.19693 +0.19094 +0.16334 +0.13221 +0.11319 +0.10977 +0.11643 +0.11490 +0.09894 +0.07779 +0.05199 +0.02125 -0.01518 -0.04413 -0.05452 -

0.04988 -0.02886 -0.00460 +0.01171 +0.02283 +0.03720 +0.06220 +0.09772 +0.13093 +0.15166 +0.15931 +0.15943 +0.15747 +0.15313 +0.14555 +0.13613 +0.12457 +0.11062 +0.09595 +0.08152 +0.06519 +0.05028 +0.04539 +0.05181 +0.06434 +0.07938 +0.09521 +0.11026 +0.12762 +0.14989 +0.17124 +0.18794 +0.19742 +0.19601 +0.18941 +0.18157 +0.16867 +0.15215 +0.13551 +0.11778 +0.09956 +0.08237 +0.06776 +0.05639 +0.04948 +0.04887 +0.05285 +0.06134 +0.07229 +0.08555 +0.10084 +0.11784 +0.13478 +0.14885 +0.16004 +0.16732 +0.16854 +0.16738 +0.16396 +0.15961 +0.15423 +0.14683 +0.13796 +0.12720 +0.11527 +0.10390 +0.09699 +0.09314 +0.09002 +0.08971 +0.09161 +0.09326 +0.09827 +0.10628 +0.11276 +0.11564 +0.11466 +0.10989 +0.10335 +0.09534 +0.08513 +0.07363 +0.06422 +0.05474 +0.04728 +0.04362 +0.04233 +0.04410 +0.05223 +0.06574 +0.08103 +0.09846 +0.11564 +0.13032 +0.14683 +0.16224 +0.17026 +0.17448 +0.17307 +0.16420 +0.15093 +0.13643 +0.12029 +0.10341 +0.08678 +0.07039 +0.05676 +0.04423 +0.03359 +0.03011 +0.03231 +0.03934 +0.04747 +0.05566 +0.06110 +0.06226 +0.06134 +0.05963 +0.05504 +0.04624 +0.03426 +0.01996 +0.00841 +0.00004 -0.00552 -0.00540 -0.00057 +0.00774 +0.01428 +0.02155 +0.03469 +0.05138 +0.06825 +0.08567 +0.10267 +0.11570 +0.12628 +0.13288 +0.13594 +0.13814 +0.13661 +0.13331 +0.12554 +0.11374 +0.09986 +0.08378 +0.06911 +0.05853 +0.05022 +0.04300 +0.03915 +0.03622 +0.03665 +0.04245 +0.05156 +0.06421 +0.07718 +0.08977 +0.09937 +0.09986 +0.09393 +0.08243 +0.06819 +0.05297 +0.03958 +0.03157 +0.02265 +0.01489 +0.01122 +0.00810 +0.00633 +0.00957 +0.01593 +0.02045 +0.02791 +0.03989 +0.05688 +0.07932 +0.10493 +0.12456 +0.14010 +0.15233 +0.15643 +0.15527 +0.15239 +0.14707 +0.13900 +0.12915 +0.12328 +0.11716 +0.10866 +0.09821 +0.08531 +0.07192 +0.05835 +0.04771 +0.04184 +0.04068 +0.04178 +0.04661 +0.05443 +0.06299 +0.07167 +0.07821 +0.08146 +0.08372 +0.08408 +0.08158 +0.07479 +0.06360 +0.05132 +0.03573 +0.02075 +0.00731 -0.00400 -0.00888 -0.00809 -0.00326 +0.00597 +0.01373 +0.02088 +0.02571 +0.02809 +0.03286 +0.03958 +0.04600 +0.05199 +0.05511 +0.05468 +0.05676 +0.06097 +0.06311 +0.06440 +0.06428 +0.06519 +0.06782 +0.07008 +0.07424 +0.07987 +0.08622 +0.09491 +0.10353 +0.11166 +0.12041 +0.12512 +0.12518 +0.12499 +0.12285 +0.11765 +0.11252 +0.10426 +0.09252 +0.07932 +0.06318 +0.04636 +0.03188 +0.02155 +0.01525 +0.01483 +0.01959 +0.02564 +0.03261 +0.04111 +0.05101 +0.06055 +0.06990 +0.07944 +0.08787 +0.09271 +0.09356 +0.09075 +0.08677 +0.08146 +0.07577 +0.07228 +0.07124 +0.07406 +0.07932 +0.08347 +0.08763 +0.08922 +0.08702 +0.08433 +0.07913 +0.07271 +0.06635 +0.05816 +0.04753 +0.03188 +0.01403 -0.00112 -0.01292 -0.02086 -0.02422 -0.02257 -0.01842 -0.01322 -0.00894 -0.00491 -0.00180 +0.00150 +0.00377 +0.00340 +0.00218 +0.00071 -0.00290 -0.00650 -0.01066 -0.01530 -0.01848 -0.01970 -0.02153 -0.02190 -0.01866 -0.01420 -0.00834 +0.00052 +0.00755 +0.01434 +0.02106 +0.02638 +0.03206 +0.03713 +0.04202 +0.04581 +0.04746 +0.05034 +0.05156 +0.05199 +0.05315 +0.05394 +0.05565 +0.05724 +0.05828 +0.05840 +0.05627 +0.05388 +0.05168 +0.04716 +0.04117 +0.03744 +0.03347 +0.02858 +0.02332 +0.02020 +0.01770 +0.01623 +0.01513 +0.01379 +0.01330 +0.01275 +0.01330 +0.01421 +0.01507 +0.01421 +0.00847 -0.00180 -0.01695 -0.03681 -0.05959 -0.07993 -0.09734 -0.11138 -0.12133 -0.12744 -0.12695 -0.12872 -0.12597 -0.11694 -0.11413 -0.10918 -0.11034 -0.11169 -0.11602 -0.12457 -0.13348 -0.14264 -0.15271 -0.16284 -0.17212 -0.18200 -0.18975 -0.19317 -0.19463 -0.19390 -0.18646 -0.17743 -0.16711 -0.15771 -0.14838 -0.13800 -0.12609 -0.11285 -0.09990 -0.08812 -0.07712 -0.06650 -0.05666 -0.04591 -

0.03565 -0.02819 -0.02227 -0.01921 -0.01775 -0.01781 -0.02025 -0.02349 -0.02465 -0.02783 -0.03064 -0.03015 -0.02880 -0.02795 -0.02673 -0.02581 -0.02648 -0.03027 -0.03601 -
0.04426 -0.05379 -0.05953 -0.06301 -0.06576 -0.06723 -0.06912 -0.07266 -0.07590 -0.07853 -0.07938 -0.07633 -0.07089 -0.06478 -0.05770 -0.05031 -0.04609 -0.04316 -0.04145 -
0.04188 -0.04298 -0.04304 -0.04267 -0.04224 -0.04096 -0.03766 -0.03424 -0.03143 -0.03021 -0.03033 -0.03082 -0.03247 -0.03314 -0.03216 -0.03131 -0.03021 -0.02874 -0.02715 -
0.02379 -0.02337 -0.02758 -0.03412 -0.04243 -0.04970 -0.05373 -0.05251 -0.04640 -0.03779 -0.02679 -0.01493 -0.00760 -0.00589 -0.00754 -0.01457 -0.02459 -0.03412 -0.04273 -
0.04939 -0.05281 -0.05336 -0.05086 -0.04603 -0.04188 -0.03974 -0.04121 -0.04487 -0.04951 -0.05581 -0.06033 -0.06295 -0.06478 -0.06595 -0.06595 -0.06656 -0.06967 -0.07517 -
0.08353 -0.09337 -0.10350 -0.11230 -0.11639 -0.11804 -0.11700 -0.11132 -0.10430 -0.09996 -0.09807 -0.09795 -0.10131 -0.10607 -0.10888 -0.10692 -0.10069 -0.09294 -0.08457 -
0.07890 -0.07572 -0.07328 -0.06931 -0.06369 -0.05819 -0.05477 -0.05465 -0.05452 -0.05367 -0.05196 -0.05043 -0.04945 -0.04897 -0.04891 -0.04927 -0.05031 -0.05171 -0.05239 -
0.05288 -0.05391 -0.05544 -0.05868 -0.06473 -0.07163 -0.07737 -0.07816 -0.07425 -0.06735 -0.06063 -0.05532 -0.05330 -0.05214 -0.05171 -0.05208 -0.05275 -0.05367 -0.05532 -
0.06045 -0.06845 -0.07725 -0.08433 -0.08671 -0.08244 -0.07151 -0.05874 -0.04921 -0.04567 -0.04774 -0.05043 -0.05019 -0.04512 -0.03595 -0.02624 -0.02001 -0.01891 -0.02300 -
0.03107 -0.04121 -0.05013 -0.05642 -0.05953 -0.06204 -0.06576 -0.06986 -0.07126 -0.07053 -0.06815 -0.06234 -0.05758 -0.05733 -0.05972 -0.06247 -0.06540 -0.06796 -0.06796 -
0.06747 -0.06912 -0.07254 -0.07890 -0.08738 -0.09599 -0.10314 -0.10692 -0.10888 -0.10967 -0.11083 -0.11034 -0.10760 -0.10051 -0.09013 -0.08048 -0.07517 -0.07541 -0.07767 -
0.07969 -0.07999 -0.07981 -0.07908 -0.07755 -0.07694 -0.07688 -0.07712 -0.07798 -0.07798 -0.07621 -0.07346 -0.07028 -0.06735 -0.06521 -0.06436 -0.06375 -0.06259 -0.05959 -
0.05746 -0.05953 -0.06405 -0.06980 -0.07615 -0.08122 -0.08372 -0.08506 -0.08647 -0.08842 -0.09160 -0.09508 -0.09715 -0.09990 -0.10241 -0.10381 -0.10503 -0.10656 -0.11150 -
0.12048 -0.13226 -0.14520 -0.15869 -0.16919 -0.17749 -0.18512 -0.19116 -0.19464 -0.19726 -0.19909 -0.19921 -0.19787 -0.19281 -0.18439 -0.17432 -0.16382 -0.15204 -0.14117 -
0.13067 -0.11853 -0.10644 -0.09575 -0.08799 -0.08189 -0.07480 -0.06637 -0.05862 -0.05117 -0.04658 -0.04646 -0.04811 -0.05031 -0.05398 -0.05905 -0.06357 -0.06808 -0.07181 -
0.07480 -0.07767 -0.07822 -0.07572 -0.07193 -0.06644 -0.05978 -0.05416 -0.04854 -0.04298 -0.03760 -0.03156 -0.02771 -0.02716 -0.02795 -0.02960 -0.03375 -0.03876 -0.04414 -
0.05013 -0.05434 -0.05501 -0.05520 -0.05721 -0.06100 -0.06369 -0.06338 -0.06137 -0.05874 -0.05795 -0.05721 -0.05587 -0.05440 -0.05159 -0.04854 -0.04952 -0.05031 -0.04842 -
0.04817 -0.04903 -0.05007 -0.05141 -0.05074 -0.04915 -0.04720 -0.04616 -0.04750 -0.05111 -0.05361 -0.05153 -0.05043 -0.05221 -0.05416 -0.05666 -0.05825 -0.05807 -0.05813 -
0.06021 -0.05202 -0.04158 -0.17456 -0.21422 -0.06192 -0.07456 -0.17035 -0.13544 -0.12646 -0.13147 -0.09331 -0.11096 -0.14820 -0.12695 -0.06564 +0.00187 +0.04025 +0.05290
+0.02478 -0.03052 -0.06998 -0.09172 -0.11108 -0.13568 -0.18219 -0.24173 -0.28935 -0.31209 -0.30307 -0.27204 -0.22929 -0.18555 -0.13178 -0.07163 -0.02258 +0.01311 +0.03829
+0.05773 +0.07589 +0.08756 +0.08206 +0.05107 +0.00187 -0.04793 -0.08818 -0.11456 -0.13153 -0.14612 -0.16211 -0.17432 -0.17615 -0.16364 -0.13910 -0.11199 -0.08604 -0.06082
-0.03870 -0.02337 -0.01335 -0.00522 -0.00009 +0.00217 +0.00486 +0.00266 -0.00535 -0.01579 -0.02679 -0.03339 -0.03596 -0.04249 -0.05068 -0.05789 -0.06748 -0.07175 -0.06662

-0.05611 -0.04573 -0.03693 -0.03101 -0.02557 -0.02050 -0.02111 -0.02404 -0.02526 -0.02862 -0.03614 -0.04665 -0.05691 -0.06387 -0.06974 -0.07768 -0.08372 -0.08543 -0.08604 -0.08366 -0.07621 -0.06882 -0.06106 -0.05025 -0.03944 -0.02875 -0.01952 -0.01616 -0.01592 -0.01127 -0.00736 -0.00370 -0.00070 -0.00168 -0.00700 -0.01860 -0.03461 -0.04964 -0.06082 -0.07193 -0.08360 -0.09404 -0.09972 -0.10082 -0.09881 -0.09734 -0.09484 -0.08965 -0.08397 -0.08152 -0.08116 -0.07792 -0.07438 -0.07090 -0.06650 -0.06332 -0.06198 -0.06271 -0.06650 -0.07157 -0.07566 -0.08048 -0.08543 -0.09026 -0.09184 -0.08794 -0.08158 -0.07291 -0.06308 -0.05398 -0.04445 -0.03516 -0.02881 -0.02465 -0.02245 -0.02245 -0.02649 -0.03565 -0.04585 -0.05257 -0.05630 -0.06228 -0.07077 -0.08110 -0.09148 -0.09874 -0.10027 -0.09703 -0.08952 -0.08030 -0.07371 -0.06845 -0.06461 -0.06167 -0.05556 -0.04787 -0.04188 -0.03865 -0.03858 -0.03913 -0.03999 -0.04176 -0.04219 -0.03999 -0.03968 -0.04048 -0.04091 -0.04225 -0.04494 -0.04708 -0.04671 -0.04463 -0.04084 -0.03553 -0.02734 -0.01531 -0.00247 +0.00626 +0.01109 +0.01555 +0.01757 +0.01812 +0.01690 +0.00804 -0.00388 -0.01482 -0.02441 -0.02954 -0.02991 -0.02911 -0.02899 -0.03125 -0.03791 -0.04543 -0.05221 -0.05325 -0.05074 -0.05001 -0.05080 -0.05459 -0.05941 -0.06412 -0.06815 -0.06900 -0.06375 -0.05722 -0.05178 -0.04824 -0.04708 -0.04744 -0.04915 -0.04879 -0.04787 -0.04763 -0.04824 -0.05025 -0.05386 -0.05893 -0.06302 -0.06442 -0.06033 -0.05337 -0.04836 -0.04164 -0.03571 -0.03321 -0.03339 -0.03522 -0.03590 -0.03498 -0.03052 -0.02337 -0.01408 -0.00425 +0.00571 +0.01238 +0.01635 +0.02209 +0.02759 +0.02937 +0.02949 +0.02943 +0.02729 +0.02338 +0.02014 +0.01971 +0.02118 +0.02368 +0.02570 +0.02533 +0.02289 +0.01916 +0.01396 +0.00822 +0.00327 -0.00302 -0.00614 -0.01005 -0.01647 -0.02343 -0.02795 -0.03186 -0.03364 -0.03315 -0.03125 -0.03052 -0.02930 -0.02692 -0.02362 -0.01867 -0.01421 -0.00804 +0.00028 +0.00694 +0.01066 +0.01250 +0.01305 +0.01317 +0.01268 +0.00938 +0.00700 +0.00663 +0.00455 +0.00174 -0.00272 -0.00785 -0.01305 -0.01763 -0.01897 -0.01683 -0.01262 -0.00822 -0.00382 -0.00180 -0.00095 +0.00150 +0.00333 +0.00584 +0.00938 +0.01024 +0.00785 +0.00522 +0.00229 -0.00027 -0.00125 -0.00082 -0.00089 -0.00205 -0.00333 -0.00321 -0.00339 -0.00125 +0.00467 +0.01231 +0.02111 +0.03010 +0.03688 +0.04092 +0.04269 +0.04477 +0.04556 +0.04391 +0.04012 +0.03450 +0.02827 +0.02166 +0.01482 +0.00632 -0.00076 -0.00431 -0.00651 -0.00828 -0.00950 -0.00932 -0.00761 -0.00547 -0.00406 -0.00364 -0.00278 -0.00119 -0.00058 +0.00058 +0.00241 +0.00241 +0.00223 +0.00174 +0.00089 +0.00095 +0.00070 +0.00046 +0.00052 +0.00021 -0.00046 -0.00174 -0.00272 -0.00321 -0.00339 -0.00260 -0.00174 +0.00040 +0.00119 +0.00168 +0.00241 +0.00394 +0.00571 +0.00742 +0.00999 +0.01048 +0.01011 +0.00993 +0.00803 +0.00370 -0.00474 -0.01421 -0.02704 -0.04402 -0.06223 -0.07921 -0.09392 -0.10626 -0.11334 -0.11865 -0.12048 -0.11865 -0.11841 -0.11615 -0.10424 -0.10015 -0.09899 -0.09887 -0.09978 -0.10363 -0.11371 -0.12158 -0.12720 -0.13037 -0.13575 -0.13770 -0.13788 -0.13947 -0.13746 -0.13520 -0.13068 -0.12488 -0.11682 -0.10675 -0.09765 -0.08855 -0.07859 -0.06784 -0.05526 -0.04341 -0.03180 -0.01952 -0.00651 +0.00657 +0.01732 +0.02270 +0.02625 +0.02729 +0.02447 +0.02050 +0.01555 +0.00883 -0.00046 -0.01073 -0.01989 -0.02704 -0.03388 -0.03969 -0.04262 -0.04476 -0.04512 -0.04671 -0.05013 -0.05312 -0.05538 -0.05728 -0.05624 -0.05422 -0.05093 -0.04604 -0.04054 -0.03443 -0.02905 -0.02264 -0.01604 -0.01140 -0.00645 -0.00254 +0.00162 +0.00443 +0.00663 +0.00938 +0.01213 +0.01500 +0.01867 +0.02038 +0.02166 +0.02227 +0.02203 +0.02282 +0.02344 +0.02509 +0.02808 +0.03126 +0.03682 +0.04342 +0.05021 +0.05608 +0.06543 +0.07478

+0.07986 +0.08169 +0.07967 +0.07319 +0.06531 +0.05589 +0.04477 +0.03615 +0.03218 +0.03034 +0.03077 +0.03523 +0.03859 +0.03945 +0.04238 +0.04373 +0.04452 +0.04568 +0.04483 +0.04385 +0.04336 +0.04165 +0.04061 +0.04098 +0.04024 +0.03798 +0.03499 +0.03358 +0.03438 +0.03493 +0.03804 +0.04183 +0.04330 +0.04495 +0.04403 +0.04281 +0.04275 +0.04122 +0.04000 +0.03872 +0.03780 +0.03713 +0.03389 +0.03114 +0.02881 +0.02509 +0.02203 +0.01934 +0.01512 +0.00907 +0.00186 -0.00523 -0.01109 -0.01702 -0.02105 -0.02447 -0.02869 -0.03278 -0.03578 -0.03785 -0.04072 -0.04109 -0.04078 -0.04072 -0.04054 -0.03944 -0.03633 -0.03101 -0.02521 -0.01971 -0.01244 -0.00523 +0.00174 +0.00895 +0.01604 +0.02099 +0.02392 +0.02692 +0.02729 +0.02521 +0.02215 +0.01995 +0.01989 +0.02026 +0.01897 +0.01665 +0.01341 +0.00981 +0.00541 +0.00345 +0.00455 +0.00871 +0.01598 +0.02875 +0.04330 +0.05675 +0.06958 +0.07955 +0.08738 +0.09270 +0.09728 +0.09948 +0.09802 +0.09361 +0.08646 +0.07607 +0.06567 +0.05907 +0.05436 +0.05210 +0.05393 +0.05968 +0.06268 +0.06262 +0.06060 +0.05559 +0.05008 +0.04550 +0.04275 +0.03988 +0.03468 +0.02747 +0.01769 +0.00852 +0.00278 +0.00131 +0.00259 +0.00754 +0.01274 +0.01616 +0.01665 +0.01506 +0.01231 +0.01127 +0.01647 +0.02405 +0.03114 +0.03676 +0.04153 +0.04470 +0.04507 +0.04617 +0.04929 +0.05332 +0.05644 +0.05766 +0.05675 +0.05430 +0.05228 +0.05241 +0.05216 +0.05155 +0.05118 +0.04996 +0.04752 +0.04611 +0.04623 +0.04941 +0.05552 +0.06213 +0.06677 +0.06891 +0.07081 +0.07313 +0.07338 +0.07136 +0.06812 +0.06286 +0.05650 +0.04953 +0.04085 +0.03217 +0.02606 +0.02276 +0.02215 +0.02478 +0.02832 +0.03169 +0.03523 +0.04006 +0.04623 +0.05216 +0.05785 +0.06408 +0.06793 +0.07001 +0.07313 +0.07545 +0.07686 +0.07766 +0.07821 +0.07943 +0.08053 +0.08004 +0.07741 +0.07405 +0.07124 +0.06867 +0.06549 +0.06341 +0.06194 +0.05944 +0.05705 +0.05724 +0.05895 +0.06158 +0.06402 +0.06488 +0.06524 +0.06555 +0.06445 +0.06133 +0.05773 +0.05308 +0.04739 +0.04238 +0.03853 +0.03609 +0.03327 +0.03089 +0.03065 +0.03217 +0.03609 +0.04189 +0.04813 +0.05504 +0.06421 +0.07496 +0.08493 +0.09398 +0.10162 +0.10639 +0.10719 +0.10480 +0.09899 +0.09111 +0.08206 +0.07350 +0.06586 +0.05895 +0.05357 +0.05033 +0.04953 +0.04959 +0.05063 +0.05241 +0.05320 +0.05241 +0.04874 +0.04220 +0.03340 +0.02301 +0.00919 -0.00706 -0.02460 -0.04329 -0.06162 -0.07774 -0.09154 -0.10143 -0.10589 -0.10711 -0.10443 -0.09844 -0.09038 -0.08177 -0.07218 -0.06162 -0.05270 -0.04347 -0.03523 -0.02954 -0.02582 -0.02209 -0.01665 -0.00865 +0.00235 +0.01482 +0.02820 +0.04257 +0.05779 +0.07270 +0.08505 +0.09441 +0.10150 +0.10529 +0.10615 +0.10346 +0.11330 +0.11991 +0.09728 +0.17098 +0.23902 +0.25738 +0.27366 +0.26374 +0.21411 +0.14737 +0.09826 +0.09649 +0.11086 +0.11556 +0.12388 +0.13929 +0.14400 +0.12939 +0.11226 +0.11202 +0.13691 +0.17759 +0.20671 +0.21197 +0.19319 +0.16101 +0.13562 +0.12828 +0.13685 +0.14841 +0.15562 +0.15978 +0.15991 +0.15269 +0.13887 +0.12241 +0.10780 +0.10119 +0.10737 +0.12847</READINGS>

<FEATURE NAME="AF_CENTRE_OFFSET_A" DT="UT_LENGTH_MM" STATUS="AF_TOLERANCE_OK">-0.0780</FEATURE>

<FEATURE NAME="AF_CENTRE_OFFSET_B" DT="UT_LENGTH_MM" STATUS="AF_TOLERANCE_OK">0.0016</FEATURE>

<FEATURE NAME="AF_BEST_RADIUS" DT="UT_LENGTH_MM" STATUS="AF_TOLERANCE_OK">99.9954</FEATURE>

<FEATURE NAME="AF_CIRCULARITY" DT="UT_LENGTH_MM">0.3933</FEATURE>

<FEATURE NAME="AF_CALCULATED_FEEDRATE" DT="UT_FEEDRATE" STATUS="AF_TOLERANCE_OK">4742.0</FEATURE>

</BALLBAR_RUN>

<BALLBAR_RUN>

<RUN_DIRECTION>CW</RUN_DIRECTION>

<RUN_TEMPERATURE>20.0</RUN_TEMPERATURE>

<START_INDEX>767</START_INDEX>

<END_INDEX>2208</END_INDEX>

<SAMPLE_RATE>200.000</SAMPLE_RATE>

<READINGS>+0.71699 +0.71337 +0.70932 +0.70668 +0.70650 +0.70613 +0.70607 +0.70717 +0.70705 +0.70349 +0.69784 +0.69116 +0.68263 +0.67263 +0.66238 +0.65349 +0.64619 +0.64011 +0.63263 +0.62147 +0.60700 +0.59111 +0.57713 +0.56457 +0.55396 +0.54519 +0.53759 +0.53054 +0.52374 +0.51657 +0.50823 +0.50051 +0.49616 +0.49211 +0.48856 +0.48439 +0.48004 +0.47869 +0.48016 +0.48121 +0.48298 +0.48648 +0.48862 +0.48856 +0.48519 +0.47900 +0.46956 +0.46037 +0.45198 +0.44463 +0.43997 +0.43605 +0.43127 +0.42453 +0.41473 +0.40689 +0.40272 +0.39856 +0.39684 +0.39758 +0.39746 +0.39672 +0.39586 +0.39378 +0.39084 +0.38870 +0.38594 +0.38251 +0.37810 +0.37106 +0.36138 +0.35220 +0.34411 +0.33713 +0.33162 +0.32758 +0.32330 +0.32030 +0.31852 +0.31779 +0.31852 +0.32121 +0.32360 +0.32458 +0.32532 +0.32452 +0.32268 +0.31932 +0.31491 +0.31166 +0.30726 +0.30083 +0.29330 +0.28283 +0.27004 +0.25878 +0.24709 +0.23638 +0.22805 +0.22004 +0.21312 +0.20633 +0.20046 +0.19666 +0.19165 +0.18736 +0.18357 +0.17880 +0.17409 +0.16754 +0.16216 +0.15769 +0.15372 +0.15066 +0.14687 +0.14418 +0.14308 +0.14075 +0.13794 +0.13341 +0.12809 +0.12136 +0.11396 +0.10681 +0.09984 +0.09299 +0.08761 +0.08315 +0.07954 +0.07758 +0.07532 +0.07294 +0.07006 +0.06536 +0.06108 +0.05698 +0.05368 +0.04903 +0.04506 +0.04188 +0.03638 +0.03167

+0.02361 +0.01193 -0.00206 -0.01844 -0.03738 -0.05845 -0.08105 -0.10163 -0.11879 -0.13338 -0.14290 -0.15016 -0.15499 -0.15828 -0.16121 -0.15676 -0.15444 -0.15950 -0.15932 -
0.16231 -0.17000 -0.18190 -0.19313 -0.20313 -0.21216 -0.22259 -0.22955 -0.23290 -0.23894 -0.24041 -0.23949 -0.23644 -0.23065 -0.22211 -0.21271 -0.20326 -0.19563 -0.18800 -
0.18031 -0.17189 -0.16329 -0.15383 -0.14491 -0.13655 -0.12831 -0.12391 -0.12562 -0.12855 -0.13264 -0.13777 -0.14345 -0.15041 -0.15706 -0.16481 -0.17549 -0.18538 -0.19343 -
0.19990 -0.20557 -0.21106 -0.21686 -0.22497 -0.23309 -0.24345 -0.25559 -0.26681 -0.27797 -0.28730 -0.29443 -0.30205 -0.30510 -0.30455 -0.30272 -0.29803 -0.29181 -0.28297 -
0.27437 -0.26821 -0.26328 -0.25907 -0.25559 -0.25303 -0.25303 -0.25334 -0.25376 -0.25651 -0.25767 -0.25870 -0.26120 -0.26602 -0.27004 -0.27382 -0.27724 -0.27901 -0.28126 -
0.28138 -0.27870 -0.27547 -0.27078 -0.26382 -0.25602 -0.24754 -0.23937 -0.23376 -0.23138 -0.23108 -0.23522 -0.24382 -0.25486 -0.26675 -0.27803 -0.28748 -0.29583 -0.30205 -
0.30498 -0.30705 -0.30949 -0.30991 -0.30894 -0.30589 -0.29833 -0.29059 -0.28388 -0.27785 -0.27730 -0.28065 -0.28675 -0.29504 -0.30345 -0.31101 -0.31771 -0.32119 -0.32155 -
0.31948 -0.31729 -0.31747 -0.31625 -0.31777 -0.31832 -0.31948 -0.32125 -0.32021 -0.31918 -0.31558 -0.31083 -0.31095 -0.31259 -0.31668 -0.32369 -0.33185 -0.34355 -0.35744 -
0.37017 -0.38357 -0.39666 -0.40732 -0.41572 -0.42181 -0.42826 -0.43684 -0.44670 -0.45406 -0.45869 -0.46027 -0.45954 -0.45558 -0.45005 -0.44627 -0.44067 -0.43581 -0.43002 -
0.42491 -0.41846 -0.41261 -0.40908 -0.40513 -0.40336 -0.40257 -0.40330 -0.40494 -0.40543 -0.40446 -0.40239 -0.39952 -0.39587 -0.39337 -0.39319 -0.39636 -0.40111 -0.40446 -
0.40586 -0.40427 -0.39952 -0.39015 -0.37876 -0.36725 -0.35738 -0.35177 -0.35238 -0.35251 -0.35348 -0.35683 -0.36097 -0.36475 -0.36822 -0.37102 -0.37401 -0.37596 -0.37772 -
0.38059 -0.38302 -0.38394 -0.38442 -0.38619 -0.38972 -0.39569 -0.40458 -0.41694 -0.43100 -0.44518 -0.45936 -0.47560 -0.49306 -0.50674 -0.51592 -0.52206 -0.52267 -0.51792 -
0.51227 -0.50272 -0.48576 -0.46678 -0.44974 -0.43398 -0.41949 -0.40768 -0.39776 -0.38771 -0.37833 -0.37255 -0.37492 -0.38229 -0.39167 -0.40409 -0.41408 -0.41888 -0.41882 -
0.41706 -0.41395 -0.40567 -0.39398 -0.38320 -0.36926 -0.35872 -0.35342 -0.35068 -0.35086 -0.35379 -0.35628 -0.35701 -0.35695 -0.35768 -0.36006 -0.36347 -0.36828 -0.37163 -
0.37255 -0.37492 -0.37986 -0.38570 -0.39009 -0.39344 -0.39399 -0.39137 -0.38802 -0.38241 -0.37608 -0.36914 -0.36311 -0.35799 -0.35269 -0.34812 -0.34769 -0.34940 -0.35080 -
0.35044 -0.35019 -0.35080 -0.35184 -0.35147 -0.35086 -0.34800 -0.34288 -0.33490 -0.32485 -0.31784 -0.31516 -0.31461 -0.31461 -0.31698 -0.32198 -0.32881 -0.33673 -0.34568 -
0.35616 -0.36640 -0.37431 -0.38034 -0.38558 -0.38954 -0.39009 -0.38820 -0.38552 -0.38168 -0.37194 -0.35854 -0.34575 -0.33539 -0.32564 -0.31674 -0.30516 -0.29199 -0.28328 -
0.27834 -0.27383 -0.27285 -0.27163 -0.26761 -0.26559 -0.26206 -0.25505 -0.24919 -0.24126 -0.22980 -0.22193 -0.21406 -0.20503 -0.19862 -0.19698 -0.20057 -0.20363 -0.20722 -
0.21198 -0.21631 -0.22168 -0.22461 -0.22760 -0.23150 -0.23193 -0.23089 -0.22650 -0.21918 -0.21064 -0.19734 -0.18099 -0.16781 -0.15322 -0.13723 -0.12508 -0.11256 -0.10120 -
0.09363 -0.08838 -0.08698 -0.08637 -0.08160 -0.07580 -0.07201 -0.06963 -0.06658 -0.06163 -0.05632 -0.05271 -0.04813 -0.04495 -0.04208 -0.04001 -0.03817 -0.03866 -0.04282 -
0.04691 -0.05094 -0.05558 -0.06169 -0.06884 -0.07433 -0.07708 -0.07910 -0.07898 -0.07427 -0.07189 -0.09846 -0.01575 +0.03057 -0.14107 -0.21644 -0.15395 -0.14577 -0.18392 -
0.21290 -0.21339 -0.16085 -0.08679 -0.03390 -0.03133 -0.09608 -0.18410 -0.23205 -0.22626 -0.20308 -0.19423 -0.20680 -0.21381 -0.19557 -0.14651 -0.08124 -0.02003 +0.02091

+0.03283 +0.02446 +0.00411 -0.02528 -0.06566 -0.11452 -0.15609 -0.17647 -0.17287 -0.16024 -0.15591 -0.16189 -0.16500 -0.15188 -0.12080 -0.08179 -0.05534 -0.02241 +0.02195 +0.06505 +0.11329 +0.16081 +0.19856 +0.22224 +0.21997 +0.19403 +0.15616 +0.11048 +0.06346 +0.02672 +0.00680 -0.01545 -0.04190 -0.05363 -0.05491 -0.04984 -0.04294 -0.03970 -0.03542 -0.02693 -0.01508 -0.00341 +0.00649 +0.01120 +0.01217 +0.01578 +0.01823 +0.01462 +0.00063 -0.02217 -0.04502 -0.06200 -0.07073 -0.07299 -0.07299 -0.07727 -0.08099 -0.07776 -0.07061 -0.06426 -0.06053 -0.05852 -0.05540 -0.05039 -0.04355 -0.03567 -0.02926 -0.02486 -0.02156 -0.01948 -0.02015 -0.02449 -0.03097 -0.03756 -0.04453 -0.05204 -0.05723 -0.06096 -0.06554 -0.06823 -0.06927 -0.06939 -0.06878 -0.06817 -0.06768 -0.06670 -0.06481 -0.06609 -0.07085 -0.07733 -0.08527 -0.09553 -0.10786 -0.12184 -0.13723 -0.15170 -0.16470 -0.17458 -0.18276 -0.18837 -0.19020 -0.18868 -0.18483 -0.17861 -0.17110 -0.16433 -0.15792 -0.15224 -0.14700 -0.14144 -0.13631 -0.13283 -0.12783 -0.12081 -0.11305 -0.10261 -0.09064 -0.07989 -0.06914 -0.05815 -0.04746 -0.03824 -0.03048 -0.02443 -0.02107 -0.02064 -0.02516 -0.03524 -0.04795 -0.06151 -0.07440 -0.08356 -0.08655 -0.08484 -0.08112 -0.07537 -0.06835 -0.06053 -0.05143 -0.04098 -0.03090 -0.02247 -0.01691 -0.01386 -0.01306 -0.01520 -0.02211 -0.03243 -0.04569 -0.06065 -0.07690 -0.09260 -0.10572 -0.11501 -0.11885 -0.11739 -0.11189 -0.10536 -0.10005 -0.09260 -0.08289 -0.07525 -0.06872 -0.06145 -0.05430 -0.04825 -0.04343 -0.04215 -0.04312 -0.04422 -0.04599 -0.04862 -0.05100 -0.05223 -0.05131 -0.04905 -0.04630 -0.04355 -0.04184 -0.04208 -0.04349 -0.04471 -0.04642 -0.04813 -0.05161 -0.05455 -0.05552 -0.05638 -0.05888 -0.06249 -0.06334 -0.06536 -0.06951 -0.07397 -0.07928 -0.08496 -0.08820 -0.08783 -0.08423 -0.07831 -0.07147 -0.06505 -0.06102 -0.05986 -0.05962 -0.05784 -0.05687 -0.05650 -0.05589 -0.05467 -0.05638 -0.06010 -0.06389 -0.06353 -0.05699 -0.04538 -0.03084 -0.01734 -0.00750 +0.00038 +0.00649 +0.01266 +0.01725 +0.01920 +0.01853 +0.01340 +0.00527 -0.00695 -0.02107 -0.03439 -0.04496 -0.05387 -0.05839 -0.05888 -0.05858 -0.05693 -0.05223 -0.04544 -0.03848 -0.03139 -0.02577 -0.02412 -0.02779 -0.03371 -0.04398 -0.05638 -0.06908 -0.08099 -0.09064 -0.09803 -0.10249 -0.10157 -0.09834 -0.09480 -0.09040 -0.08588 -0.08093 -0.07513 -0.06786 -0.06023 -0.05412 -0.04856 -0.04545 -0.04514 -0.04538 -0.04673 -0.05015 -0.05210 -0.05253 -0.05223 -0.04905 -0.04459 -0.04062 -0.03500 -0.02748 -0.01960 -0.01013 -0.00164 +0.00362 +0.00643 +0.00796 +0.00808 +0.00826 +0.00777 +0.00288 -0.00750 -0.02107 -0.03628 -0.05174 -0.06524 -0.07391 -0.07764 -0.07629 -0.07025 -0.06169 -0.05162 -0.04141 -0.03628 -0.03201 -0.02712 -0.02266 -0.02040 -0.01948 -0.01930 -0.01954 -0.02034 -0.02028 -0.01936 -0.01826 -0.01630 -0.01465 -0.01417 -0.01423 -0.01484 -0.01478 -0.01239 -0.00891 -0.00561 -0.00347 -0.00604 -0.01465 -0.02773 -0.04221 -0.05351 -0.06066 -0.06322 -0.06304 -0.06096 -0.05626 -0.05119 -0.04673 -0.04166 -0.03860 -0.03695 -0.03488 -0.03384 -0.03347 -0.03353 -0.03421 -0.03494 -0.03640 -0.03848 -0.04190 -0.04636 -0.05033 -0.05351 -0.05217 -0.04703 -0.03812 -0.02663 -0.01643 -0.00775 -0.00243 +0.00093 +0.00197 +0.00160 +0.00154 +0.00123 +0.00093 +0.00203 +0.00392 +0.00539 +0.00722 +0.00936 +0.01181 +0.01504 +0.01834 +0.02109 +0.02274 +0.02647 +0.02776 +0.02696 +0.02592 +0.02378 +0.02030 +0.01688 +0.01187 +0.00924 +0.00673 +0.00521 +0.00447 +0.00514 +0.00667 +0.00887 +0.01266 +0.01553 +0.01712 +0.01572 +0.01181 +0.00319 -0.01050 -0.02816 -0.04960 -0.07055 -0.08961 -0.10536 -0.11659 -0.12264 -0.12166 -0.11965 -0.11409 -0.10365 -0.09944 -0.09327 -0.09364 -0.09394 -0.09699 -0.10536 -0.11440 -0.12349 -0.13418 -0.14700 -0.15682 -0.16464 -0.17239 -

0.17550 -0.17452 -0.17263 -0.16634 -0.15731 -0.14584 -0.13418 -0.12166 -0.10866 -0.09455 -0.07990 -0.06762 -0.05571 -0.04312 -0.02999 -0.01526 -0.00084 +0.01291 +0.02128
+0.02647 +0.02898 +0.02843 +0.02733 +0.02488 +0.02372 +0.02287 +0.02158 +0.02103 +0.02097 +0.02079 +0.02146 +0.02244 +0.02354 +0.02476 +0.02397 +0.02079 +0.01682
+0.01077 +0.00465 -0.00158 -0.00787 -0.01447 -0.02040 -0.02498 -0.02810 -0.02907 -0.02663 -0.02132 -0.01478 -0.00659 +0.00117 +0.00844 +0.01743 +0.02751 +0.03558 +0.04285
+0.04799 +0.05086 +0.05337 +0.05520 +0.05416 +0.05220 +0.04884 +0.04242 +0.03527 +0.02879 +0.02604 +0.02629 +0.02953 +0.03735 +0.04493 +0.04939 +0.05514 +0.06052
+0.06590 +0.07183 +0.07513 +0.07696 +0.07703 +0.07770 +0.08008 +0.08296 +0.08338 +0.07923 +0.07360 +0.06853 +0.06584 +0.06504 +0.06688 +0.07061 +0.07434 +0.07605
+0.07440 +0.07054 +0.06675 +0.06535 +0.06724 +0.07061 +0.07311 +0.07684 +0.08075 +0.08344 +0.08558 +0.08626 +0.08277 +0.07721 +0.07183 +0.06541 +0.05948 +0.05398
+0.04952 +0.04365 +0.03674 +0.03081 +0.02598 +0.02219 +0.02140 +0.02342 +0.02452 +0.02696 +0.02934 +0.03032 +0.03173 +0.03497 +0.03784 +0.03980 +0.03961 +0.03943
+0.04077 +0.04059 +0.04132 +0.04395 +0.04573 +0.04780 +0.05367 +0.06192 +0.07134 +0.08112 +0.08748 +0.09286 +0.09567 +0.09592 +0.09402 +0.08901 +0.08149 +0.07507
+0.07012 +0.06749 +0.06724 +0.06822 +0.07177 +0.07238 +0.06840 +0.06205 +0.05593 +0.05324 +0.05398 +0.06021 +0.07085 +0.08069 +0.08803 +0.09353 +0.09561 +0.09647
+0.09708 +0.09592 +0.09298 +0.08736 +0.07917 +0.06810 +0.05526 +0.04316 +0.03405 +0.03014 +0.02751 +0.02733 +0.03313 +0.04096 +0.05000 +0.06082 +0.07207 +0.08173
+0.08650 +0.09115 +0.09207 +0.08999 +0.08956 +0.09078 +0.09133 +0.09042 +0.08895 +0.08656 +0.07910 +0.06792 +0.05783 +0.04933 +0.04181 +0.03717 +0.03417 +0.03625
+0.04304 +0.04933 +0.05410 +0.05844 +0.06156 +0.06174 +0.06119 +0.06162 +0.06211 +0.06223 +0.06235 +0.06272 +0.06205 +0.05923 +0.05196 +0.04065 +0.02916 +0.02183
+0.01798 +0.01957 +0.02641 +0.03564 +0.04640 +0.05532 +0.06070 +0.06785 +0.07568 +0.08051 +0.08418 +0.08785 +0.09207 +0.09745 +0.10332 +0.10637 +0.10870 +0.11139
+0.11090 +0.10894 +0.10815 +0.10705 +0.10527 +0.10460 +0.10387 +0.10252 +0.10252 +0.10246 +0.10185 +0.10234 +0.09971 +0.09408 +0.08950 +0.08632 +0.08259 +0.07990
+0.07574 +0.07232 +0.07024 +0.06657 +0.06315 +0.06211 +0.06223 +0.06382 +0.06626 +0.06730 +0.06639 +0.06767 +0.07085 +0.07415 +0.07825 +0.08033 +0.08265 +0.08791
+0.09402 +0.09787 +0.10362 +0.10839 +0.10760 +0.10533 +0.10270 +0.09934 +0.09861 +0.09916 +0.09720 +0.09616 +0.09586 +0.09317 +0.09372 +0.09787 +0.10081 +0.10283
+0.10466 +0.10521 +0.10533 +0.10595 +0.10558 +0.10472 +0.10136 +0.09561 +0.08980 +0.08503 +0.08106 +0.08002 +0.08155 +0.08638 +0.09237 +0.09775 +0.10270 +0.10607
+0.10974 +0.11585 +0.12258 +0.12655 +0.12723 +0.12625 +0.12215 +0.11585 +0.11041 +0.10387 +0.09793 +0.09274 +0.08571 +0.07971 +0.07721 +0.07562 +0.07702 +0.08204
+0.08687 +0.09066 +0.09439 +0.09873 +0.10283 +0.10650 +0.10851 +0.10906 +0.10870 +0.10711 +0.10289 +0.10056 +0.09738 +0.09280 +0.09151 +0.09219 +0.09225 +0.09262
+0.09310 +0.09421 +0.09757 +0.10264 +0.10680 +0.10845 +0.10900 +0.10833 +0.10772 +0.10790 +0.10613 +0.10197 +0.09683 +0.09194 +0.08785 +0.08467 +0.07696 +0.06718
+0.06810 -0.02333 -0.13699 -0.14804 -0.15780 -0.15225 -0.05272 +0.03167 +0.03472 +0.03448 +0.04585 +0.05013 +0.06058 +0.06828 +0.06681 +0.06437 +0.07103 +0.07250
+0.04774 -0.00072 -0.05290 -0.08545 -0.07898 -0.04197 -0.00971 -0.00512 -0.02345 -0.03964 -0.02871 +0.00270 +0.04426 +0.08797 +0.11989 +0.14264 +0.16741 +0.19054 +0.20326

+0.20387 +0.19861 +0.19158 +0.18632 +0.17677 +0.14435 +0.09072 +0.03270 -0.01398 -0.03830 -0.04447 -0.04783 -0.05950 -0.07367 -0.08039 -0.07587 -0.05950 -0.03427 -0.00476 +0.02659 +0.05893 +0.08748 +0.10643 +0.11390 +0.10980 +0.10008 +0.09225 +0.08247 +0.06419 +0.03973 +0.01351 -0.00940 -0.02315 -0.02908 -0.03250 -0.03311 -0.02791 -0.01655 +0.00166 +0.02690 +0.05300 +0.07543 +0.09329 +0.10619 +0.11359 +0.11634 +0.11481 +0.10894 +0.09971 +0.08797 +0.07274 +0.05422 +0.03417 +0.01492 +0.00092 -0.00574 -0.00678 -0.00427 +0.00025 +0.00728 +0.01865 +0.03539 +0.05465 +0.07134 +0.08350 +0.09078 +0.09445 +0.09696 +0.09720 +0.09549 +0.09261 +0.08901 +0.08161 +0.07122 +0.06125 +0.05129 +0.04303 +0.04102 +0.04426 +0.05006 +0.05654 +0.06528 +0.07409 +0.08118 +0.08858 +0.09610 +0.10203 +0.10582 +0.10631 +0.10166 +0.09659 +0.09200 +0.08497 +0.07745 +0.07213 +0.06632 +0.05850 +0.04939 +0.03918 +0.03203 +0.03136 +0.03399 +0.03710 +0.04016 +0.04279 +0.04340 +0.04309 +0.04426 +0.04652 +0.04805 +0.05153 +0.05630 +0.06260 +0.07268 +0.08393 +0.09261 +0.09916 +0.10656 +0.11506 +0.12129 +0.12319 +0.12172 +0.11628 +0.10821 +0.09885 +0.08931 +0.08112 +0.07562 +0.07073 +0.06223 +0.05159 +0.03894 +0.02708 +0.01926 +0.01596 +0.01602 +0.01816 +0.02225 +0.02476 +0.02396 +0.02329 +0.02531 +0.02861 +0.03136 +0.03478 +0.03759 +0.03686 +0.03576 +0.03698 +0.03845 +0.04083 +0.04627 +0.05147 +0.05471 +0.05428 +0.05012 +0.04328 +0.03753 +0.03484 +0.03350 +0.03148 +0.03008 +0.02946 +0.03142 +0.03765 +0.04835 +0.06058 +0.07079 +0.07549 +0.07482 +0.07024 +0.06094 +0.05294 +0.05012 +0.04964 +0.04627 +0.04089 +0.03405 +0.02531 +0.01565 +0.00917 +0.00752 +0.00740 +0.00789 +0.00801 +0.00435 -0.00268 -0.01007 -0.01374 -0.01148 -0.00500 +0.00319 +0.01345 +0.02244 +0.02751 +0.02818 +0.02898 +0.03063 +0.03258 +0.03753 +0.04010 +0.03833 +0.03344 +0.02616 +0.02048 +0.01822 +0.01852 +0.01932 +0.02384 +0.02898 +0.03179 +0.03332 +0.03234 +0.03093 +0.03081 +0.03142 +0.03313 +0.03362 +0.03454 +0.03729 +0.03979 +0.04144 +0.04407 +0.04682 +0.04847 +0.05141 +0.05373 +0.05385 +0.05324 +0.05263 +0.05000 +0.04603 +0.04242 +0.03949 +0.03772 +0.03710 +0.03668 +0.03270 +0.02494 +0.01589 +0.00673 -0.00195 -0.00996 -0.01741 -0.02297 -0.02798 -0.03262 -0.03629 -0.03769 -0.03769 -0.03556 -0.03030 -0.02260 -0.01631 -0.01112 -0.00549 -0.00061 +0.00422 +0.00893 +0.01400 +0.01889 +0.02121 +0.02060 +0.01718 +0.01235 +0.00838 +0.00520 +0.00202 +0.00074 +0.00184 +0.00410 +0.00734 +0.01174 +0.01644 +0.02078 +0.02421 +0.02763 +0.03117 +0.03405 +0.03539 +0.03453 +0.03068 +0.02531 +0.02029 +0.01638 +0.01540 +0.01846 +0.02256 +0.02555 +0.02744 +0.02940 +0.03099 +0.03007 +0.02916 +0.02836 +0.02671 +0.02512 +0.02494 +0.02604 +0.02867 +0.03185 +0.03649 +0.04108 +0.04566 +0.05055 +0.05391 +0.05727 +0.06076 +0.06504 +0.06883 +0.06944 +0.06754 +0.06351 +0.05758 +0.05037 +0.04156 +0.03374 +0.02714 +0.02072 +0.01443 +0.00978 +0.00709 +0.00630 +0.00526 +0.00520 +0.00526 +0.00569 +0.00697 +0.00728 +0.00868 +0.01058 +0.01033 +0.00960 +0.00801 +0.00679 +0.00734 +0.00862 +0.00935 +0.01131 +0.01382 +0.01455 +0.01522 +0.01388 +0.01229 +0.01052 +0.00972 +0.00935 +0.01033 +0.01009 +0.01119 +0.01229 +0.01345 +0.01467 +0.01534 +0.01473 +0.01296 +0.01052 +0.00783 +0.00398 -0.00219 -0.01051 -0.02108 -0.03360 -0.04851 -0.06543 -0.08271 -0.09724 -0.11019 -0.11916 -0.12503 -0.12814 -0.12716 -0.12674 -0.12582 -0.11483 -0.11214 -0.11220 -0.11227 -0.11428 -0.12051 -0.13315 -0.14267 -0.14914 -0.15353 -0.16098 -0.16238 -0.16214 -0.16159 -0.15756 -0.15243 -0.14633 -0.13882 -0.12979 -0.11996 -0.10940 -0.09950 -0.08882 -0.07697 -0.06378 -0.05113 -

0.03843 -0.02511 -0.01142 +0.00214 +0.01449 +0.02060 +0.02390 +0.02518 +0.02341 +0.02115 +0.01816 +0.01259 +0.00587 -0.00158 -0.00910 -0.01545 -0.02095 -0.02749 -0.03329 -0.03953 -0.04661 -0.05315 -0.06091 -0.06860 -0.07483 -0.07990 -0.08167 -0.08161 -0.08124 -0.07966 -0.07740 -0.07398 -0.06885 -0.06256 -0.05608 -0.05003 -0.04246 -0.03439 -0.02688 -0.01986 -0.01503 -0.01204 -0.00971 -0.00996 -0.01295 -0.01588 -0.01876 -0.02151 -0.02309 -0.02346 -0.02352 -0.02261 -0.01937 -0.01509 -0.00751 +0.00269 +0.01400 +0.02433 +0.03802 +0.05140 +0.06033 +0.06650 +0.06956 +0.06852 +0.06473 +0.05904 +0.04926 +0.03777 +0.02610 +0.01504 +0.00532 -0.00116 -0.00495 -0.00592 -0.00269 +0.00031 +0.00465 +0.00996 +0.01528 +0.02292 +0.03093 +0.03832 +0.04315 +0.04462 +0.04321 +0.04003 +0.03551 +0.02940 +0.02634 +0.02439 +0.02274 +0.02280 +0.02261 +0.02219 +0.02188 +0.02292 +0.02543 +0.02628 +0.02677 +0.02763 +0.02824 +0.02952 +0.02769 +0.02689 +0.02732 +0.02567 +0.02280 +0.01772 +0.00972 -0.00110 -0.01313 -0.02627 -0.03873 -0.05010 -0.05810 -0.06280 -0.06665 -0.07111 -0.07490 -0.07716 -0.07966 -0.07850 -0.07697 -0.07648 -0.07447 -0.06952 -0.06054 -0.05205 -0.04594 -0.04185 -0.03770 -0.03403 -0.03043 -0.02670 -0.02328 -0.01692 -0.00953 -0.00427 -0.00177 -0.00201 -0.00299 -0.00568 -0.00953 -0.01381 -0.01815 -0.02138 -0.02535 -0.02633 -0.02248 -0.01808 -0.01344 -0.00745 -0.00244 +0.00355 +0.01064 +0.01290 +0.01510 +0.01730 +0.01772 +0.01559 +0.01222 +0.00972 +0.00624 +0.00129 -0.00317 -0.00568 -0.00819 -0.01112 -0.01344 -0.01405 -0.01191 -0.00837 -0.01063 -0.01680 -0.02590 -0.04075 -0.05987 -0.07819 -0.09364 -0.10891 -0.12118 -0.12619 -0.13101 -0.13602 -0.13571 -0.12930 -0.11892 -0.10598 -0.09108 -0.07545 -0.05895 -0.04527 -0.03666 -0.02835 -0.02114 -0.01796 -0.01735 -0.01967 -0.02535 -0.03091 -0.03611 -0.04044 -0.04472 -0.04790 -0.04845 -0.04820 -0.04967 -0.04888 -0.04405 -0.03342 -0.02138 -0.01246 -0.00739 -0.00611 -0.00617 -0.00837 -0.01265 -0.01717 -0.02316 -0.03250 -0.04393 -0.05450 -0.05847 -0.05590 -0.05266 -0.04967 -0.04924 -0.05395 -0.05871 -0.06378 -0.07068 -0.07203 -0.06964 -0.06751 -0.06470 -0.05993 -0.05474 -0.04637 -0.03715 -0.03220 -0.03104 -0.03269 -0.03794 -0.04185 -0.04216 -0.04259 -0.04228 -0.04039 -0.03978 -0.04228 -0.04448 -0.04332 -0.04045 -0.03929 -0.04100 -0.04454 -0.04839 -0.05230 -0.05615 -0.05975 -0.06622 -0.07771 -0.09340 -0.10775 -0.11661 -0.12241 -0.12527 -0.12802 -0.13236 -0.13651 -0.13858 -0.13889 -0.13474 -0.12607 -0.11679 -0.10855 -0.09762 -0.08180 -0.06537 -0.05157 -0.04277 -0.03971 -0.04075 -0.04503 -0.05083 -0.05486 -0.05615 -0.05578 -0.05462 -0.05205 -0.04753 -0.04020 -0.03189 -0.02359 -0.01638 -0.01509 -0.01772 -0.01803 -0.01943 -0.02395 -0.03275 -0.04478 -0.05560 -0.06610 -0.07789 -0.08821 -0.09414 -0.09792 -0.09957 -0.09584 -0.08711 -0.07343 -0.05804 -0.04130 -0.02328 -0.00519 +0.01033 +0.02396 +0.03551 +0.04034 +0.03679 +0.02927 +0.02188 +0.01546 +0.00856 +0.00134 -0.00501 -0.00929 -0.01118 -0.01167 -0.01173 -0.00923 -0.00428 -0.00049 +0.00409 +0.00959 +0.01082 +0.01021 +0.01008 +0.00727 +0.00049 -0.00807 -0.01949 -0.03177 -0.04368 -0.05651 -0.06659 -0.08162 -0.11612 -0.02994 -0.03073 -0.20571 -0.24433 -0.18448 -0.18680 -0.21749 -0.24487 -0.23237 -0.16861 -0.09664 -0.04570 -0.04680 -0.11239 -0.18979 -0.22176 -0.20882 -0.19180 -0.19571 -0.21444 -0.22530 -0.20913 -0.16208 -0.10360 -0.05187 -0.02053 -0.01332 -0.02212 -0.03990 -0.06329 -0.09908 -0.14225 -0.17398 -0.18527 -0.17838 -0.16654 -0.16501 -0.17362 -0.17771 -0.16361 -0.13608 -0.10494 -0.07862 -0.05853 -0.05034 -0.03330 +0.00764 +0.05079 +0.08765 +0.11817 +0.13676 +0.14581 +0.14104 +0.12184 +0.09921 +0.07738 +0.05776 +0.04376 +0.03808 +0.02628 +0.00287 -0.01069 -0.01283 -

0.01002 -0.00794 -0.01576 -0.02688 -0.03489 -0.03745 -0.03360 -0.02682 -0.02261 -0.02242 -0.02261 -0.02181 -0.02249 -0.02829 -0.03715 -0.04540 -0.05205 -0.05450 -0.05541 -
0.05841 -0.06390 -0.07074 -0.07471 -0.07490 -0.07704 -0.08430 -0.09139 -0.09548 -0.09518 -0.08882 -0.07936 -0.06769 -0.05480 -0.04308 -0.03190 -0.02212 -0.01668 -0.01326 -
0.01186 -0.01412 -0.02017 -0.02768 -0.03636 -0.04583 -0.05389 -0.06250 -0.07044 -0.07685 -0.08241 -0.08540 -0.08504 -0.08137 -0.07753 -0.07569 -0.07649 -0.08052 -0.08724 -
0.09743 -0.10812 -0.11942 -0.13169 -0.14384 -0.15610 -0.16819 -0.17911 -0.18668 -0.19046 -0.19107 -0.18924 -0.18424 -0.17673 -0.16862 -0.15958 -0.15024 -0.14127 -0.13291 -
0.12546 -0.11899 -0.11325 -0.10733 -0.09939 -0.09060 -0.08131 -0.07154 -0.06128 -0.05187 -0.04509 -0.03966 -0.03660 -0.03568 -0.03721 -0.04216 -0.05102 -0.06110 -0.07020 -
0.07863 -0.08363 -0.08327 -0.08101 -0.07869 -0.07545 -0.07197 -0.06763 -0.05957 -0.04931 -0.04014 -0.03110 -0.02401 -0.02065 -0.01907 -0.01986 -0.02640 -0.03703 -0.04802 -
0.05932 -0.07093 -0.08156 -0.09157 -0.09994 -0.10458 -0.10568 -0.10336 -0.09817 -0.09389 -0.08956 -0.08162 -0.07496 -0.07056 -0.06702 -0.06305 -0.06055 -0.05932 -0.05951 -
0.06201 -0.06360 -0.06470 -0.06531 -0.06574 -0.06519 -0.06220 -0.05865 -0.05639 -0.05432 -0.05132 -0.05096 -0.05230 -0.05199 -0.05193 -0.05181 -0.05108 -0.04949 -0.04735 -
0.04546 -0.04405 -0.04595 -0.04821 -0.05145 -0.05816 -0.06635 -0.07618 -0.08663 -0.09493 -0.10006 -0.10214 -0.10146 -0.09811 -0.09365 -0.08815 -0.08321 -0.07863 -0.07404 -
0.06928 -0.06384 -0.05786 -0.04973 -0.04247 -0.03666 -0.03544 -0.03666 -0.03721 -0.03660 -0.03458 -0.03196 -0.02878 -0.02469 -0.02151 -0.01839 -0.01424 -0.01173 -0.01039 -
0.00850 -0.00678 -0.00733 -0.01063 -0.01736 -0.02444 -0.03348 -0.04332 -0.05035 -0.05603 -0.05865 -0.05676 -0.05377 -0.05065 -0.04723 -0.04381 -0.04088 -0.04002 -0.03856 -
0.03752 -0.03953 -0.04485 -0.05279 -0.06116 -0.06849 -0.07533 -0.07863 -0.08027 -0.08198 -0.08162 -0.08107 -0.08040 -0.08046 -0.07911 -0.07740 -0.07685 -0.07631 -0.07527 -
0.07093 -0.06360 -0.05536 -0.04888 -0.04503 -0.04112 -0.03929 -0.03947 -0.04039 -0.04204 -0.04125 -0.03617 -0.02817 -0.01760 -0.00734 -0.00043 +0.00415 +0.00666 +0.00660
+0.00586 +0.00311 -0.00153 -0.00703 -0.01504 -0.02444 -0.03434 -0.04552 -0.05597 -0.06317 -0.06672 -0.06794 -0.06757 -0.06446 -0.05774 -0.04900 -0.03892 -0.02750 -0.01693 -
0.01247 -0.01406 -0.01901 -0.02567 -0.03104 -0.03336 -0.03104 -0.02591 -0.01785 -0.00789 -0.00153 +0.00122 +0.00006 -0.00593 -0.01394 -0.02237 -0.03031 -0.03489 -0.03776 -
0.04015 -0.04424 -0.04937 -0.05383 -0.05896 -0.06434 -0.06824 -0.06806 -0.06464 -0.05902 -0.05377 -0.04858 -0.04534 -0.04271 -0.04039 -0.04167 -0.04485 -0.04845 -0.05242 -
0.05548 -0.05627 -0.05603 -0.05438 -0.05181 -0.04949 -0.04790 -0.04473 -0.04149 -0.03844 -0.03226 -0.02628 -0.02047 -0.01442 -0.00935 -0.00501 -0.00312 -0.00226 -0.00294 -
0.00453 -0.00501 -0.00514 -0.00391 -0.00037 +0.00372 +0.00892 +0.01393 +0.01692 +0.01949 +0.02071 +0.02175 +0.02249 +0.02108 +0.01906 +0.01674 +0.01320 +0.00971
+0.00507 +0.00207 +0.00000 -0.00049 +0.00000 +0.00097 +0.00391 +0.00733 +0.01094 +0.01411 +0.01729 +0.01772 +0.01411 +0.00586 -0.00807 -0.02707 -0.05010 -0.07307 -
0.09316 -0.10898 -0.12058 -0.12784 -0.12815 -0.12510 -0.12149 -0.10934 -0.10177 -0.09444 -0.09200 -0.09328 -0.09719 -0.10745 -0.11826 -0.12943 -0.14054 -0.15159 -0.16349 -
0.17271 -0.17960 -0.18558 -0.18796 -0.18662 -0.18070 -0.17088 -0.15775 -0.14469 -0.13053 -0.11679 -0.10244 -0.08699 -0.07380 -0.06183 -0.04937 -0.03721 -0.02414 -0.01076
+0.00244 +0.01185 +0.01741 +0.02230 +0.02548 +0.02689 +0.02799 +0.02780 +0.02823 +0.02774 +0.02652 +0.02591 +0.02524 +0.02481 +0.02352 +0.02163 +0.01888 +0.01515

+0.01112 +0.00831 +0.00330 -0.00184 -0.00673 -0.01241 -0.01815 -0.02377 -0.02811 -0.03110 -0.03165 -0.02952 -0.02506 -0.01858 -0.00990 -0.00111 +0.00678 +0.01497 +0.02218 +0.02725 +0.03208 +0.03618 +0.03948 +0.04259 +0.04498 +0.04406 +0.04186 +0.03856 +0.03294 +0.02756 +0.02358 +0.02407 +0.02615 +0.02945 +0.03538 +0.04070 +0.04559 +0.05066 +0.05567 +0.06185 +0.06686 +0.07010 +0.07096 +0.06815 +0.06509 +0.06552 +0.06863 +0.07145 +0.07261 +0.07298 +0.07261 +0.07261 +0.07261 +0.07426 +0.07762 +0.08019 +0.08092 +0.08019 +0.07664 +0.07114 +0.06692 +0.06539 +0.06607 +0.06747 +0.07151 +0.07719 +0.08282 +0.08716 +0.08961 +0.08899 +0.08478 +0.07768 +0.06863 +0.05940 +0.05231 +0.04877 +0.04834 +0.04742 +0.04651 +0.04363 +0.03630 +0.02927 +0.02303 +0.01759 +0.01729 +0.02108 +0.02585 +0.03184 +0.03746 +0.03923 +0.03764 +0.03495 +0.03269 +0.03214 +0.03153 +0.03355 +0.03990 +0.04742 +0.05586 +0.06374 +0.06980 +0.07371 +0.07512 +0.07414 +0.07640 +0.08105 +0.08374 +0.08416 +0.08245 +0.07982 +0.07738 +0.07420 +0.07212 +0.07126 +0.07157 +0.07181 +0.07065 +0.06888 +0.06594 +0.06338 +0.06099 +0.05879 +0.06069 +0.06533 +0.06925 +0.07518 +0.08478 +0.09541 +0.10330 +0.10660 +0.10587 +0.10067 +0.09058 +0.07799 +0.06313 +0.04926 +0.03795 +0.02762 +0.02224 +0.02334 +0.02695 +0.03379 +0.04198 +0.04993 +0.05574 +0.05953 +0.06332 +0.06729 +0.07401 +0.08245 +0.09187 +0.09988 +0.10312 +0.09859 +0.08997 +0.08233 +0.07615 +0.07035 +0.06619 +0.05940 +0.05011 +0.04082 +0.03294 +0.02676 +0.02609 +0.02964 +0.03239 +0.03532 +0.04137 +0.04785 +0.05219 +0.05653 +0.06069 +0.06191 +0.06222 +0.06148 +0.05971 +0.05745 +0.05561 +0.05207 +0.04382 +0.03483 +0.02829 +0.02285 +0.01796 +0.01888 +0.02438 +0.03110 +0.03850 +0.04553 +0.05372 +0.06215 +0.07145 +0.08111 +0.08850 +0.09670 +0.10550 +0.11156 +0.11492 +0.11694 +0.11565 +0.11070 +0.10581 +0.10006 +0.09554 +0.09346 +0.09327 +0.09523 +0.09792 +0.09908 +0.09921 +0.09841 +0.09706 +0.09615 +0.09553 +0.09358 +0.09223 +0.09028 +0.08428 +0.07530 +0.06857 +0.06252 +0.05580 +0.05225 +0.05194 +0.05335 +0.05965 +0.06704 +0.07334 +0.07817 +0.08239 +0.08422 +0.08483 +0.08490 +0.08276 +0.08233 +0.08331 +0.08447 +0.08691 +0.08979 +0.09333 +0.09645 +0.09939 +0.10348 +0.10984 +0.11602 +0.12116 +0.12483 +0.12513 +0.12262 +0.12024 +0.11761 +0.11547 +0.11321 +0.10728 +0.10098 +0.09725 +0.09419 +0.09083 +0.08899 +0.08887 +0.08832 +0.08655 +0.08447 +0.08300 +0.08337 +0.08600 +0.09015 +0.09700 +0.10348 +0.10770 +0.11052 +0.11125 +0.11198 +0.11492 +0.11682 +0.11755 +0.11859 +0.11871 +0.11853 +0.11865 +0.11804 +0.11590 +0.11431 +0.11149 +0.10520 +0.09884 +0.09284 +0.08838 +0.08716 +0.08655 +0.08697 +0.08679 +0.08587 +0.08642 +0.08801 +0.08850 +0.08856 +0.08844 +0.08887 +0.08954 +0.08899 +0.08967 +0.09119 +0.09242 +0.09407 +0.09712 +0.09823 +0.09743 +0.09688 +0.09810 +0.10092 +0.10471 +0.10868 +0.11070 +0.11156 +0.11205 +0.11272 +0.11315 +0.11229 +0.11094 +0.10825 +0.10605 +0.10410 +0.10098 +0.09590 +0.08471 +0.09132 +0.01882 -0.10562 -0.13767 -0.14518 -0.15684 -0.07619 +0.02120 +0.03556 +0.03165 +0.04375 +0.04614 +0.05848 +0.06839 +0.06949 +0.06814 +0.07303 +0.07860 +0.06062 +0.01790 -0.03343 -0.07203 -0.07374 -0.04155 -0.00709 +0.00232 -0.01864 -0.04082 -0.04473 -0.03172 -0.00245 +0.02914 +0.05176 +0.06649 +0.08159 +0.09547 +0.09547 +0.08331 +0.06741 +0.05586 +0.05451 +0.05451 +0.04247 +0.01790 -0.00514 -0.01522 -0.00838 +0.00947</READINGS>

<FEATURE NAME="AF_CENTRE_OFFSET_A" DT="UT_LENGTH_MM" STATUS="AF_TOLERANCE_OK">-0.0510</FEATURE>

<FEATURE NAME="AF_CENTRE_OFFSET_B" DT="UT_LENGTH_MM" STATUS="AF_TOLERANCE_OK">-0.0394</FEATURE>

<FEATURE NAME="AF_BEST_RADIUS" DT="UT_LENGTH_MM" STATUS="AF_TOLERANCE_OK">100.0051</FEATURE>

<FEATURE NAME="AF_CIRCULARITY" DT="UT_LENGTH_MM">0.4244</FEATURE>

<FEATURE NAME="AF_CALCULATED_FEEDRATE" DT="UT_FEEDRATE" STATUS="AF_TOLERANCE_OK">5228.7</FEATURE>

</BALLBAR_RUN>

</RUN_RESULTS>

<ANALYSIS NAME="RENISHAW_DIAGNOSTICS" version="2.0">

<FEATURE NAME="AF_CALCULATED_FEEDRATE" DT="UT_FEEDRATE">4985.4</FEATURE>

<FEATURE NAME="AF_CYCLIC_PITCH_A" DT="UT_LENGTH_MM">10.1600</FEATURE>

<FEATURE NAME="AF_CYCLIC_PITCH_B" DT="UT_LENGTH_MM">31.7500</FEATURE>

<FEATURE NAME="AF_CYCLIC_ERROR_A_FWD" DT="UT_LENGTH_UM">84.1</FEATURE>

<FEATURE NAME="AF_CYCLIC_ERROR_A_REV" DT="UT_LENGTH_UM">45.9</FEATURE>

<FEATURE NAME="AF_CYCLIC_ERROR_B_FWD" DT="UT_LENGTH_UM">81.1</FEATURE>

<FEATURE NAME="AF_CYCLIC_ERROR_B_REV" DT="UT_LENGTH_UM">111.8</FEATURE>

<FEATURE NAME="AF_CYCLIC_PHASE_A_FWD" DT="UT_ANGLE">235.30054</FEATURE>

<FEATURE NAME="AF_CYCLIC_PHASE_A_REV" DT="UT_ANGLE">59.15147</FEATURE>

<FEATURE NAME="AF_CYCLIC_PHASE_B_FWD" DT="UT_ANGLE">321.47537</FEATURE>

<FEATURE NAME="AF_CYCLIC_PHASE_B_REV" DT="UT_ANGLE">190.14136</FEATURE>

<FEATURE NAME="AF_CENTRE_OFFSET_A" DT="UT_LENGTH_UM">-63.0</FEATURE>

<FEATURE NAME="AF_CENTRE_OFFSET_B" DT="UT_LENGTH_UM">-15.6</FEATURE>

<FEATURE NAME="AF_GLOBAL_SCALE" DT="UT_LENGTH_UM">-0.6</FEATURE>

<FEATURE NAME="AF_SCALE_MISMATCH" DT="UT_LENGTH_UM">-9.5</FEATURE>

<FEATURE NAME="AF_STRAIGHTNESS_A" DT="UT_LENGTH_UM">-16.4</FEATURE>

<FEATURE NAME="AF_STRAIGHTNESS_B" DT="UT_LENGTH_UM">-13.5</FEATURE>

<FEATURE NAME="AF_SQUARENESS" DT="UT_SQUARENESS">-401.3</FEATURE>

<FEATURE NAME="AF_SCALE_ERROR_A" DT="UT_SCALE_ERROR">-30.2</FEATURE>

<FEATURE NAME="AF_SCALE_ERROR_B" DT="UT_SCALE_ERROR">17.2</FEATURE>

<FEATURE NAME="AF_BEST_RADIUS" DT="UT_LENGTH_MM">99.9994</FEATURE>

<FEATURE NAME="AF_BACKLASH_A_GLOBAL" DT="UT_LENGTH_UM">6.7</FEATURE>

<FEATURE NAME="AF_BACKLASH_B_GLOBAL" DT="UT_LENGTH_UM">-66.1</FEATURE>

<FEATURE NAME="AF_LATERAL_PLAY_A_GLOBAL" DT="UT_LENGTH_UM">-21.3</FEATURE>

<FEATURE NAME="AF_LATERAL_PLAY_B_GLOBAL" DT="UT_LENGTH_UM">73.3</FEATURE>

<FEATURE NAME="AF_SERVO_MISMATCH" DT="UT_SERVO_MISMATCH">-0.59</FEATURE>

<FEATURE NAME="AF_SIN_4_THETA" DT="UT_LENGTH_UM">30.4</FEATURE>

<FEATURE NAME="AF_BACKLASH_A_MISMATCH" DT="UT_LENGTH_UM">-23.2</FEATURE>

<FEATURE NAME="AF_BACKLASH_B_MISMATCH" DT="UT_LENGTH_UM">26.3</FEATURE>

<FEATURE NAME="AF_LATERAL_PLAY_A_MISMATCH" DT="UT_LENGTH_UM">-67.0</FEATURE>

<FEATURE NAME="AF_LATERAL_PLAY_B_MISMATCH" DT="UT_LENGTH_UM">71.6</FEATURE>

<FEATURE NAME="AF_OFFSET_A_SHIFT" DT="UT_LENGTH_UM">-43.8</FEATURE>

<FEATURE NAME="AF_OFFSET_B_SHIFT" DT="UT_LENGTH_UM">5.9</FEATURE>

<FEATURE NAME="AF_RADIUS_CHANGE" DT="UT_LENGTH_UM">-66.4</FEATURE>

<FEATURE NAME="AF_BACKLASH_A_POS" DT="UT_LENGTH_UM">-16.5</FEATURE>

<FEATURE NAME="AF_BACKLASH_A_NEG" DT="UT_LENGTH_UM">29.9</FEATURE>

<FEATURE NAME="AF_BACKLASH_B_POS" DT="UT_LENGTH_UM">-39.8</FEATURE>

<FEATURE NAME="AF_BACKLASH_B_NEG" DT="UT_LENGTH_UM">-92.3</FEATURE>

<FEATURE NAME="AF_LATERAL_PLAY_A_POS" DT="UT_LENGTH_UM">-88.3</FEATURE>

<FEATURE NAME="AF_LATERAL_PLAY_A_NEG" DT="UT_LENGTH_UM">45.7</FEATURE>

<FEATURE NAME="AF_LATERAL_PLAY_B_POS" DT="UT_LENGTH_UM">144.9</FEATURE>

<FEATURE NAME="AF_LATERAL_PLAY_B_NEG" DT="UT_LENGTH_UM">1.7</FEATURE>

<FEATURE NAME="AF_REVERSAL_SPIKES_A_POS" DT="UT_LENGTH_UM">103.2</FEATURE>

<FEATURE NAME="AF_REVERSAL_SPIKES_B_POS" DT="UT_LENGTH_UM">-71.6</FEATURE>

<FEATURE NAME="AF_REVERSAL_SPIKES_A_NEG" DT="UT_LENGTH_UM">27.9</FEATURE>

<FEATURE NAME="AF_REVERSAL_SPIKES_B_NEG" DT="UT_LENGTH_UM">-15.5</FEATURE>

<FEATURE NAME="AF_CIRCULARITY" DT="UT_LENGTH_UM">437.9</FEATURE>

<FEATURE NAME="AF_POSITIONAL_TOLERANCE" DT="UT_LENGTH_UM">508.9</FEATURE>

<FEATURE NAME="RMS" DT="UT_LENGTH_UM">42.2</FEATURE>

<FEATURE NAME="AF_CYCLIC_ERROR_B_RANK" DT="UT_RANKING">1</FEATURE>

<FEATURE NAME="AF_CYCLIC_ERROR_B_PERC" DT="UT_PERCENT">14</FEATURE>

<FEATURE NAME="AF_CYCLIC_ERROR_B_IC" DT="UT_LENGTH_UM">106.7</FEATURE>

<FEATURE NAME="AF_REVERSAL_SPIKES_A_RANK" DT="UT_RANKING">2</FEATURE>

<FEATURE NAME="AF_REVERSAL_SPIKES_A_PERC" DT="UT_PERCENT">14</FEATURE>

<FEATURE NAME="AF_REVERSAL_SPIKES_A_IC" DT="UT_LENGTH_UM">103.2</FEATURE>

<FEATURE NAME="AF_LATERAL_PLAY_B_RANK" DT="UT_RANKING">3</FEATURE>

<FEATURE NAME="AF_LATERAL_PLAY_B_PERC" DT="UT_PERCENT">13</FEATURE>

<FEATURE NAME="AF_LATERAL_PLAY_B_IC" DT="UT_LENGTH_UM">94.5</FEATURE>

<FEATURE NAME="AF_BACKLASH_B_RANK" DT="UT_RANKING">4</FEATURE>

<FEATURE NAME="AF_BACKLASH_B_PERC" DT="UT_PERCENT">12</FEATURE>

<FEATURE NAME="AF_BACKLASH_B_IC" DT="UT_LENGTH_UM">92.3</FEATURE>

<FEATURE NAME="AF_CYCLIC_ERROR_A_RANK" DT="UT_RANKING">5</FEATURE>

<FEATURE NAME="AF_CYCLIC_ERROR_A_PERC" DT="UT_PERCENT">11</FEATURE>

<FEATURE NAME="AF_CYCLIC_ERROR_A_IC" DT="UT_LENGTH_UM">82.0</FEATURE>

<FEATURE NAME="AF_REVERSAL_SPIKES_B_RANK" DT="UT_RANKING">6</FEATURE>

<FEATURE NAME="AF_REVERSAL_SPIKES_B_PERC" DT="UT_PERCENT">10</FEATURE>

<FEATURE NAME="AF_REVERSAL_SPIKES_B_IC" DT="UT_LENGTH_UM">71.6</FEATURE>

<FEATURE NAME="AF_SERVO_MISMATCH_RANK" DT="UT_RANKING">7</FEATURE>

<FEATURE NAME="AF_SERVO_MISMATCH_PERC" DT="UT_PERCENT">7</FEATURE>

<FEATURE NAME="AF_SERVO_MISMATCH_IC" DT="UT_LENGTH_UM">49.4</FEATURE>

<FEATURE NAME="AF_LATERAL_PLAY_A_RANK" DT="UT_RANKING">8</FEATURE>

<FEATURE NAME="AF_LATERAL_PLAY_A_PERC" DT="UT_PERCENT">7</FEATURE>

<FEATURE NAME="AF_LATERAL_PLAY_A_IC" DT="UT_LENGTH_UM">49.3</FEATURE>

<FEATURE NAME="AF_SQUARENESS_RANK" DT="UT_RANKING">9</FEATURE>

<FEATURE NAME="AF_SQUARENESS_PERC" DT="UT_PERCENT">5</FEATURE>

<FEATURE NAME="AF_SQUARENESS_IC" DT="UT_LENGTH_UM">40.1</FEATURE>

<FEATURE NAME="AF_BACKLASH_A_RANK" DT="UT_RANKING">10</FEATURE>

<FEATURE NAME="AF_BACKLASH_A_PERC" DT="UT_PERCENT">4</FEATURE>

<FEATURE NAME="AF_BACKLASH_A_IC" DT="UT_LENGTH_UM">29.9</FEATURE>

<FEATURE NAME="AF_STRAIGHTNESS_A_RANK" DT="UT_RANKING">11</FEATURE>

<FEATURE NAME="AF_STRAIGHTNESS_A_PERC" DT="UT_PERCENT">1</FEATURE>

<FEATURE NAME="AF_STRAIGHTNESS_A_IC" DT="UT_LENGTH_UM">8.2</FEATURE>

<FEATURE NAME="AF_STRAIGHTNESS_B_RANK" DT="UT_RANKING">12</FEATURE>

<FEATURE NAME="AF_STRAIGHTNESS_B_PERC" DT="UT_PERCENT">1</FEATURE>

<FEATURE NAME="AF_STRAIGHTNESS_B_IC" DT="UT_LENGTH_UM">6.7</FEATURE>

<FEATURE NAME="AF_SCALE_MISMATCH_RANK" DT="UT_RANKING">13</FEATURE>

<FEATURE NAME="AF_SCALE_MISMATCH_PERC" DT="UT_PERCENT">1</FEATURE>

<FEATURE NAME="AF_SCALE_MISMATCH_IC" DT="UT_LENGTH_UM">4.7</FEATURE>

<SIMULATION NAME="AF_SIMULATED_READINGS_CCW">-0.08854 -0.08915 -0.08976 -0.09038 -0.09100 -0.09163 -0.09226 -0.09290 -0.09354 -0.09366 -0.09371 -0.09377 -0.09383 -0.09389 -0.09395 -0.09402 -0.09409 -0.09415 -0.09422 -0.09429 -0.09436 -0.09443 -0.09451 -0.09458 -0.09465 -0.09473 -0.09480 -0.09487 -0.09494 -0.09468 -0.08283 -0.08231 -0.08180 -0.08127 -0.08074 -0.08020 -0.07966 -0.07910 -0.07853 -0.07795 -0.07735 -0.07674 -0.07610 -0.07545 -0.07477 -0.07407 -0.07334 -0.07258 -0.07179 -0.07097 -0.07012 -0.06923 -0.06830 -0.06734 -0.06634 -0.06531 -0.06423 -0.06313 -0.06198 -0.06080 -0.05959 -0.05834 -0.05707 -0.05578 -0.05446 -0.05313 -0.05179 -0.05045 -0.04910 -0.04777 -0.04644 -0.04514 -0.04387 -0.04263 -0.04144 -0.04029 -0.03921 -0.03820 -0.03726 -0.03640 -0.03564 -0.03497 -0.03440 -0.03393 -0.03358 -0.03333 -0.03320 -0.03318 -0.03328 -0.03349 -0.03381 -0.03422 -0.03473 -0.03533 -0.03600 -0.03673 -0.03752 -0.03835 -0.03920 -0.04006 -0.04091 -0.04174 -0.04252 -0.04325 -0.04390 -0.04446 -0.04491 -0.04525 -0.04544 -0.04549 -0.04539 -0.04514 -0.04473 -0.04416 -0.04344 -0.04258 -0.04157 -0.04043 -0.03918 -0.03784 -0.03643 -0.03498 -0.03350 -0.03204 -0.03061 -0.02924 -0.02797 -0.02682 -0.02583 -0.02501 -0.02438 -0.02397 -0.02378 -0.02383 -0.02412 -0.02465 -0.02541 -0.02642 -0.02763 -0.02902 -0.03056 -0.03223 -0.03398 -0.03578 -0.03758 -0.03935 -0.04103 -0.04258 -0.04395 -0.04513 -0.04606 -0.04673 -0.04710 -0.04718 -0.04695 -0.04641 -0.04555 -0.04441 -0.04301 -0.04140 -0.03961 -0.03769 -0.03568 -0.03365 -0.03164 -0.02971 -0.02793 -0.02634 -0.02498 -0.02389 -0.02309 -0.02260 -0.02243 -0.02258 -0.02303 -0.02377 -0.02476 -0.02595 -0.02729 -0.02872 -0.03017 -0.03159 -0.03291 -0.03408 -0.03504 -0.03573 -0.03612 -0.03618 -0.03591 -0.03531 -0.03440 -0.03319 -0.03173 -0.03007 -0.02826 -0.02638 -0.02450 -0.02269 -0.02103 -0.01958 -0.01839 -0.01752 -0.01701 -0.01686 -0.01710 -0.01774 -0.01873 -0.02002 -0.02157 -0.02330 -0.02515 -0.02704 -0.02889 -0.03063 -0.03218 -0.03346 -0.03443 -0.03507 -0.03535 -0.03528 -0.03487 -0.03415 -0.03317 -0.03198 -0.03067 -0.02931 -0.02799 -0.02679 -0.02578 -0.02502 -0.02457 -0.02445 -0.02470 -0.02531 -0.02628 -0.02756 -0.02908 -0.03078 -0.03258 -0.03441 -0.03617 -0.03780 -0.03921 -0.04034 -0.04112 -0.04155 -0.04161 -0.04132 -0.04071 -0.03982 -0.03872 -0.03748 -0.03616 -0.03487 -0.03368 -0.03268 -0.03191 -0.03143 -0.03126 -0.03141 -0.03187 -0.03262 -0.03360 -0.03477 -0.03605 -0.03734 -0.03858 -0.03969 -0.04060 -0.04125 -0.04161 -0.04166 -0.04138 -0.04078 -0.03990 -0.03881 -0.03756 -0.03623 -0.03488 -0.03360 -0.03245 -0.03148 -0.03074 -0.03029 -0.03011 -0.03020 -0.03052 -0.03104 -0.03170 -0.03245 -0.03321 -0.03392 -0.03452 -0.03493 -0.03513 -0.03509 -0.03480 -0.03427 -0.03351 -0.03256 -0.03148 -0.03030 -0.02910 -0.02793 -0.02686 -0.02593 -0.02517 -0.02461 -0.02425 -0.02410 -0.02412 -0.02430 -0.02459 -0.02493 -0.02529 -0.02561 -0.02584 -0.02595 -0.02591 -0.02571 -0.02535 -0.02482 -0.02415 -0.02337 -0.02252 -0.02164 -0.02076 -0.01993 -0.01918 -0.01852 -0.01798 -0.01703 +0.00580 +0.00591 +0.00594 +0.00589 +0.00581 +0.00572 +0.00564 +0.00561 +0.00564 +0.00574 +0.00593 +0.00619 +0.00652 +0.00691 +0.00734 +0.00779 +0.00823 +0.00866 +0.00906 +0.00941 +0.00971 +0.00997 +0.01018 +0.01036 +0.01051 +0.01065 +0.01080 +0.01096 +0.01016 -0.00037 -0.00421 -0.01135 -0.01846 -0.02555 -0.03265 -0.03975 -0.05237 -0.06415 -0.07545 -0.08679 -0.09817 -0.10714 -0.11051 -0.11388 -0.11722 -0.12052 -0.12376 -0.12692 -0.12999 -0.13296 -0.13584 -0.13784 -0.13744 -0.13605 -0.13217 -0.12670 -0.12128 -0.11593 -0.11070 -0.10558 -0.10059 -0.09572 -0.09096 -0.08629 -0.08167 -0.07708 -0.07247 -0.06780 -0.06302 -0.05811 -0.05304 -0.03317 -0.00890 -0.00739 -0.00574 -0.00399 -

0.00217 -0.00033 +0.00146 +0.00317 +0.00473 +0.00613 +0.00732 +0.00829 +0.00906 +0.00962 +0.01002 +0.01028 +0.01046 +0.01063 +0.01085 +0.01117 +0.01165 +0.01232
+0.01322 +0.01435 +0.01572 +0.01733 +0.01911 +0.02102 +0.02300 +0.02497 +0.02688 +0.02865 +0.03023 +0.03156 +0.03261 +0.03334 +0.03376 +0.03389 +0.03378 +0.03346
+0.03301 +0.03249 +0.03198 +0.03154 +0.03125 +0.03118 +0.03136 +0.03182 +0.03256 +0.03357 +0.03482 +0.03626 +0.03784 +0.03947 +0.04108 +0.04258 +0.04390 +0.04496
+0.04572 +0.04612 +0.04616 +0.04582 +0.04513 +0.04411 +0.04281 +0.04130 +0.03968 +0.03801 +0.03639 +0.03488 +0.03357 +0.03250 +0.03173 +0.03127 +0.03116 +0.03135
+0.03182 +0.03250 +0.03335 +0.03428 +0.03522 +0.03609 +0.03681 +0.03729 +0.03747 +0.03731 +0.03678 +0.03588 +0.03460 +0.03298 +0.03106 +0.02890 +0.02655 +0.02411
+0.02166 +0.01928 +0.01707 +0.01508 +0.01338 +0.01202 +0.01103 +0.01043 +0.01022 +0.01041 +0.01093 +0.01173 +0.01276 +0.01394 +0.01521 +0.01648 +0.01769 +0.01876
+0.01964 +0.02025 +0.02058 +0.02061 +0.02033 +0.01974 +0.01889 +0.01779 +0.01650 +0.01508 +0.01359 +0.01211 +0.01071 +0.00946 +0.00843 +0.00766 +0.00721 +0.00712
+0.00740 +0.00808 +0.00916 +0.01063 +0.01244 +0.01456 +0.01693 +0.01950 +0.02221 +0.02499 +0.02778 +0.03051 +0.03310 +0.03552 +0.03770 +0.03961 +0.04121 +0.04250
+0.04345 +0.04406 +0.04436 +0.04435 +0.04404 +0.04351 +0.04277 +0.04189 +0.04091 +0.03988 +0.03885 +0.03786 +0.03695 +0.03618 +0.03557 +0.03515 +0.03493 +0.03493
+0.03514 +0.03556 +0.03619 +0.03700 +0.03797 +0.03907 +0.04028 +0.04154 +0.04283 +0.04409 +0.04530 +0.04642 +0.04741 +0.04824 +0.04888 +0.04929 +0.04947 +0.04940
+0.04907 +0.04849 +0.04765 +0.04657 +0.04525 +0.04371 +0.04198 +0.04006 +0.03801 +0.03584 +0.03360 +0.03131 +0.02901 +0.02673 +0.02450 +0.02236 +0.02033 +0.01846
+0.01675 +0.01523 +0.01391 +0.01282 +0.01196 +0.01134 +0.01096 +0.01083 +0.01096 +0.01133 +0.01194 +0.01276 +0.01380 +0.01502 +0.01643 +0.01800 +0.01971 +0.02154
+0.02348 +0.02550 +0.02758 +0.02969 +0.03183 +0.03397 +0.03609 +0.03818 +0.04021 +0.04219 +0.04407 +0.04587 +0.04756 +0.04914 +0.05060 +0.05193 +0.05314 +0.05421
+0.05515 +0.05594 +0.05659 +0.05711 +0.05749 +0.05774 +0.05786 +0.05787 +0.05776 +0.05754 +0.05722 +0.05679 +0.05628 +0.05569 +0.05503 +0.05430 +0.05352 +0.05270
+0.05183 +0.05094 +0.05002 +0.04909 +0.04816 +0.04723 +0.04630 +0.04539 +0.04451 +0.04365 +0.04282 +0.04204 +0.04130 +0.04061 +0.03997 +0.03939 +0.03887 +0.03856
+0.03557 +0.03053 +0.02556 +0.02065 +0.01580 +0.01103 +0.00631 +0.00166 -0.00294 -0.00747 -0.01195 -0.01638 -0.02076 -0.02509 -0.02937 -0.03362 -0.03783 -0.04201 -0.04616
-0.05028 -0.05439 -0.05848 -0.06256 -0.06663 -0.07070 -0.07477 -0.07884 -0.07379 -0.05326 -0.03275 -0.01226 +0.00822 +0.02868 +0.04911 +0.06953 +0.08959 +0.10185 +0.09892
+0.09597 +0.10110 +0.11834 +0.13859 +0.15634 +0.18514 +0.19495 +0.19805 +0.19910 +0.19530 +0.19058 +0.18585 +0.18110 +0.17633 +0.17156 +0.16678 +0.16200 +0.15722
+0.15243 +0.14765 +0.14288 +0.13811 +0.13334 +0.12859 +0.12385 +0.11912 +0.11440 +0.10970 +0.10501 +0.10033 +0.09567 +0.09102 +0.08639 +0.09237 +0.09304 +0.09318
+0.09334 +0.09350 +0.09367 +0.09384 +0.09403 +0.09421 +0.09440 +0.09459 +0.09478 +0.09497 +0.09516 +0.09535 +0.09554 +0.09572 +0.09590 +0.09609 +0.09626 +0.09644
+0.09662 +0.09680 +0.09698 +0.09716 +0.09735 +0.09754 +0.09775 +0.09795 +0.09817 +0.09841 +0.09865 +0.09891 +0.09919 +0.09949 +0.09980 +0.10013 +0.10047 +0.10084
+0.10122 +0.10162 +0.10203 +0.10246 +0.10289 +0.10333 +0.10378 +0.10422 +0.10466 +0.10508 +0.10549 +0.10588 +0.10624 +0.10656 +0.10685 +0.10709 +0.10729 +0.10742

+0.10750 +0.10751 +0.10746 +0.10732 +0.10712 +0.10684 +0.10648 +0.10606 +0.10556 +0.10499 +0.10437 +0.10369 +0.10296 +0.10219 +0.10140 +0.10059 +0.09978 +0.09898 +0.09821 +0.09748 +0.09680 +0.09619 +0.09567 +0.09525 +0.09494 +0.09475 +0.09470 +0.09478 +0.09501 +0.09538 +0.09590 +0.09656 +0.09736 +0.09829 +0.09934 +0.10048 +0.10169 +0.10296 +0.10426 +0.10555 +0.10682 +0.10804 +0.10915 +0.11014 +0.11097 +0.11162 +0.11205 +0.11225 +0.11218 +0.11183 +0.11119 +0.11025 +0.10898 +0.10741 +0.10556 +0.10342 +0.10104 +0.09843 +0.09563 +0.09268 +0.08962 +0.08649 +0.08335 +0.08025 +0.07723 +0.07435 +0.07166 +0.06918 +0.06696 +0.06503 +0.06342 +0.06215 +0.06123 +0.06065 +0.06038 +0.06043 +0.06075 +0.06131 +0.06207 +0.06298 +0.06400 +0.06506 +0.06611 +0.06709 +0.06795 +0.06865 +0.06915 +0.06941 +0.06942 +0.06917 +0.06864 +0.06784 +0.06680 +0.06557 +0.06417 +0.06267 +0.06111 +0.05956 +0.05808 +0.05673 +0.05558 +0.05469 +0.05410 +0.05385 +0.05395 +0.05443 +0.05528 +0.05649 +0.05804 +0.05987 +0.06195 +0.06420 +0.06655 +0.06892 +0.07124 +0.07342 +0.07541 +0.07714 +0.07855 +0.07960 +0.08024 +0.08048 +0.08033 +0.07982 +0.07898 +0.07786 +0.07653 +0.07505 +0.07351 +0.07197 +0.07053 +0.06926 +0.06821 +0.06743 +0.06696 +0.06682 +0.06699 +0.06748 +0.06824 +0.06922 +0.07037 +0.07160 +0.07282 +0.07396 +0.07494 +0.07569 +0.07614 +0.07624 +0.07595 +0.07525 +0.07413 +0.07265 +0.07083 +0.06874 +0.06644 +0.06402 +0.06155 +0.05913 +0.05683 +0.05474 +0.05292 +0.05141 +0.05024 +0.04940 +0.04889 +0.04868 +0.04872 +0.04895 +0.04929 +0.04966 +0.04998 +0.05016 +0.05015 +0.04987 +0.04930 +0.04840 +0.04717 +0.04564 +0.04381 +0.04175 +0.03953 +0.03724 +0.03494 +0.03271 +0.03064 +0.02878 +0.02720 +0.02592 +0.02498 +0.02438 +0.02407 +0.02403 +0.02420 +0.02452 +0.02491 +0.02531 +0.02563 +0.02582 +0.02581 +0.02556 +0.02506 +0.02430 +0.02330 +0.02209 +0.02071 +0.01922 +0.01768 +0.01616 +0.01473 +0.01345 +0.01238 +0.01154 +0.01095 +0.01063 +0.01056 +0.01071 +0.01105 +0.01153 +0.01208 +0.01265 +0.01318 +0.01361 +0.01390 +0.01403 +0.01397 +0.01373 +0.01330 +0.01271 +0.01199 +0.01120 +0.01039 +0.00959 +0.00886 +0.00824 +0.00776 +0.00743 +0.04825 +0.04753 +0.04685 +0.04630 +0.04585 +0.04547 +0.04511 +0.04473 +0.04431 +0.04381 +0.04321 +0.04250 +0.04167 +0.04073 +0.03970 +0.03860 +0.03745 +0.03628 +0.03511 +0.03398 +0.03291 +0.03190 +0.03098 +0.03013 +0.02935 +0.02864 +0.02797 +0.02733 +0.02671 +0.01818 +0.00857 -0.00107 -0.01074 -0.02047 -0.03023 -0.04002 -0.05676 -0.07408 -0.08232 -0.10597 -0.09060 -0.10481 -0.11170 -0.11857 -0.12541 -0.13224 -0.13908 -0.14593 -0.15282 -0.15974 -0.16672 -0.17299 -0.17565 -0.17665 -0.17469 -0.16960 -0.16451 -0.15939 -0.15422 -0.14898 -0.14365 -0.13823 -0.13272 -0.12712 -0.12146 -0.11574 -0.11001 -0.10429 -0.09860 -0.09300 -0.08750 -0.07321 -0.04735 -0.04749 -0.04776 -0.04815 -0.04862 -0.04915 -0.04968 -0.05019 -0.05061 -0.05092 -0.05109 -0.05109 -0.05092 -0.05057 -0.05007 -0.04944 -0.04871 -0.04794 -0.04718 -0.04649 -0.04591 -0.04549 -0.04527 -0.04526 -0.04548 -0.04591 -0.04656 -0.04737 -0.04829 -0.04927 -0.05024 -0.05114 -0.05192 -0.05253 -0.05291 -0.05306 -0.05294 -0.05258 -0.05200 -0.05125 -0.05039 -0.04946 -0.04855 -0.04773 -0.04705 -0.04658 -0.04638 -0.04648 -0.04688 -0.04757 -0.04853 -0.04971 -0.05108 -0.05254 -0.05405 -0.05550 -0.05683 -0.05796 -0.05885 -0.05945 -0.05975 -0.05973 -0.05942 -0.05885 -0.05807 -0.05714 -0.05614 -0.05516 -0.05428 -0.05357 -0.05310 -0.05292 -0.05306 -0.05355 -0.05438 -0.05556 -0.05702 -0.05871 -0.06054 -0.06244 -0.06434 -0.06614 -0.06777 -0.06916 -0.07025 -0.07097 -0.07134 -0.07134 -0.07100 -0.07036 -0.06946 -0.06837 -0.06717 -0.06593 -0.06475 -

0.06372 -0.06289 -0.06234 -0.06210 -0.06221 -0.06268 -0.06349 -0.06464 -0.06606 -0.06772 -0.06954 -0.07144 -0.07332 -0.07512 -0.07675 -0.07814 -0.07925 -0.08001 -0.08040 -0.08037 -0.07997 -0.07920 -0.07811 -0.07675 -0.07519 -0.07349 -0.07172 -0.06998 -0.06832 -0.06684 -0.06560 -0.06464 -0.06400 -0.06371 -0.06377 -0.06418 -0.06492 -0.06596 -0.06726 -0.06875 -0.07037 -0.07205 -0.07371 -0.07529 -0.07673 -0.07797 -0.07896 -0.07965 -0.08001 -0.08003 -0.07970 -0.07906 -0.07811 -0.07690 -0.07547 -0.07387 -0.07215 -0.07038 -0.06863 -0.06696 -0.06543 -0.06409 -0.06298 -0.06216 -0.06165 -0.06146 -0.06162 -0.06212 -0.06297 -0.06412 -0.06556 -0.06723 -0.06910 -0.07113 -0.07325 -0.07543 -0.07760 -0.07971 -0.08172 -0.08357 -0.08523 -0.08668 -0.08788 -0.08882 -0.08949 -0.08988 -0.09000 -0.08985 -0.08945 -0.08883 -0.08803 -0.08708 -0.08600 -0.08484 -0.08364 -0.08243 -0.08124 -0.08013 -0.07912 -0.07824 -0.07751 -0.07695 -0.07658 -0.07641 -0.07645 -0.07669 -0.07714 -0.07780 -0.07863 -0.07964 -0.08079 -0.08206 -0.08344 -0.08489 -0.08639 -0.08792 -0.08945 -0.09094 -0.09238 -0.09373 -0.09499 -0.09613 -0.09714 -0.09799 -0.09868 -0.09921 -0.09955 -0.09970 -0.09968 -0.09948 -0.09911 -0.09858 -0.09789 -0.09706 -0.09610 -0.09502 -0.09383 -0.09255 -0.09121 -0.08981 -0.08837 -0.08692 -0.08546 -0.08401 -0.08258 -0.08119 -0.07986 -0.07859 -0.07740 -0.07629 -0.07526 -0.07434 -0.07351 -0.07279 -0.07218 -0.07168 -0.07129 -0.07102 -0.07085 -0.07078 -0.07082 -0.07095 -0.07117 -0.07148 -0.07186 -0.07233 -0.07286 -0.07345 -0.07409 -0.07478 -0.07550 -0.07626 -0.07704 -0.07784 -0.07865 -0.07946 -0.08027 -0.08108 -0.08187 -0.08265 -0.08340 -0.08414 -0.08484 -0.08551 -0.08615 -0.08675 -0.08732 -0.08784 -0.08832 -0.08877 -0.08917 -0.06583 -0.05833 -0.05923 -0.06009 -0.06091 -0.06169 -0.06244 -0.06315 -0.06383 -0.06448 -0.06510 -0.06570 -0.06627 -0.06682 -0.06735 -0.06786 -0.06836 -0.06884 -0.06930 -0.06976 -0.07063 -0.07166 -0.07268 -0.07370 -0.07471 -0.07572 -0.07673 -0.07773 -0.07874 -0.07974 -0.08074 -0.08174 -0.08275 -0.08375 -0.08475 -0.08575 -0.08534 -0.05847 -0.00610 +0.04627 +0.04132 -0.02387</SIMULATION>

<SIMULATION NAME="AF_SIMULATED_READINGS_CW">-0.06288 -0.06402 -0.06521 -0.06644 -0.06771 -0.06904 -0.07043 -0.07187 -0.00556 -0.00659 -0.00286 +0.00080 +0.00440 +0.01115 +0.01819 +0.02069 +0.01867 +0.01568 +0.00460 -0.00654 -0.01523 -0.02289 -0.02951 -0.03367 -0.03786 -0.04045 -0.04289 -0.04534 -0.04780 -0.05025 -0.05270 -0.05514 -0.05755 -0.05995 -0.06231 -0.06463 -0.06692 -0.06916 -0.07135 -0.07349 -0.07558 -0.07760 -0.07957 -0.08147 -0.08330 -0.05833 -0.05854 -0.05868 -0.05876 -0.05876 -0.05870 -0.05857 -0.05838 -0.05812 -0.05780 -0.05742 -0.05697 -0.05648 -0.05592 -0.05531 -0.05465 -0.05394 -0.05318 -0.05238 -0.05154 -0.05065 -0.04973 -0.04877 -0.04778 -0.04676 -0.04571 -0.04464 -0.04356 -0.04246 -0.04136 -0.04025 -0.03914 -0.03805 -0.03698 -0.03593 -0.03492 -0.03395 -0.03304 -0.03220 -0.03143 -0.03076 -0.03018 -0.02971 -0.02938 -0.02917 -0.02912 -0.02922 -0.02949 -0.02994 -0.03057 -0.03140 -0.03241 -0.03362 -0.03503 -0.03662 -0.03840 -0.04035 -0.04245 -0.04470 -0.04708 -0.04955 -0.05210 -0.05469 -0.05728 -0.05986 -0.06237 -0.06477 -0.06704 -0.06912 -0.07097 -0.07255 -0.07383 -0.07476 -0.07531 -0.07546 -0.07518 -0.07444 -0.07323 -0.07155 -0.06940 -0.06679 -0.06373 -0.06025 -0.05638 -0.05217 -0.04765 -0.04290 -0.03796 -0.03291 -0.02782 -0.02277 -0.01782 -0.01306 -0.00855 -0.00438 -0.00060 +0.00273 +0.00555 +0.00783 +0.00954

+0.01065 +0.01117 +0.01110 +0.01045 +0.00927 +0.00760 +0.00549 +0.00302 +0.00025 -0.00273 -0.00584 -0.00900 -0.01211 -0.01509 -0.01788 -0.02039 -0.02258 -0.02440 -0.02582 -0.02682 -0.02741 -0.02761 -0.02746 -0.02699 -0.02628 -0.02540 -0.02444 -0.02349 -0.02264 -0.02199 -0.02162 -0.02163 -0.02207 -0.02301 -0.02449 -0.02653 -0.02913 -0.03227 -0.03591 -0.03999 -0.04441 -0.04910 -0.05393 -0.05879 -0.06354 -0.06807 -0.07226 -0.07598 -0.07914 -0.08167 -0.08349 -0.08457 -0.08491 -0.08451 -0.08342 -0.08170 -0.07945 -0.07676 -0.07377 -0.07059 -0.06737 -0.06423 -0.06129 -0.05866 -0.05644 -0.05469 -0.05344 -0.05271 -0.05249 -0.05271 -0.05332 -0.05422 -0.05528 -0.05639 -0.05741 -0.05820 -0.05863 -0.05860 -0.05799 -0.05676 -0.05484 -0.05223 -0.04896 -0.04507 -0.04064 -0.03578 -0.03062 -0.02531 -0.01998 -0.01480 -0.00989 -0.00539 -0.00140 +0.00201 +0.00479 +0.00693 +0.00846 +0.00942 +0.00992 +0.01004 +0.00993 +0.00973 +0.00957 +0.00959 +0.00992 +0.01066 +0.01189 +0.01366 +0.01597 +0.01881 +0.02213 +0.02583 +0.02981 +0.03394 +0.03808 +0.04209 +0.04584 +0.04919 +0.05205 +0.05435 +0.05604 +0.05711 +0.05760 +0.05755 +0.05706 +0.05623 +0.05520 +0.05409 +0.05305 +0.05220 +0.05164 +0.05147 +0.05174 +0.05248 +0.05368 +0.05531 +0.05730 +0.05956 +0.06197 +0.06442 +0.06678 +0.06894 +0.07080 +0.07226 +0.07328 +0.07382 +0.07388 +0.07350 +0.07273 +0.07166 +0.07038 +0.06901 +0.06765 +0.06643 +0.06542 +0.06471 +0.06436 +0.06439 +0.06481 +0.06559 +0.06668 +0.06802 +0.06951 +0.07106 +0.07259 +0.07400 +0.07521 +0.07617 +0.07682 +0.07716 +0.07719 +0.07694 +0.07645 +0.07579 +0.07503 +0.07425 +0.07352 +0.07293 +0.07252 +0.07234 +0.07242 +0.07277 +0.07337 +0.01578 +0.01963 +0.02361 +0.02767 +0.03175 +0.03579 +0.03974 +0.04356 +0.04724 +0.05076 +0.05413 +0.05737 +0.06051 +0.06358 +0.06662 +0.06967 +0.07278 +0.07598 +0.07929 +0.08272 +0.08628 +0.08996 +0.08652 +0.08317 +0.07987 +0.07661 +0.07335 +0.07008 +0.06677 +0.06344 +0.06006 +0.05666 +0.05324 +0.07293 +0.06950 +0.06608 +0.06271 +0.07180 +0.07583 +0.07992 +0.08405 +0.08822 +0.09238 +0.09652 +0.10060 +0.10315 +0.09244 +0.08162 +0.07066 +0.05959 +0.05562 +0.05158 +0.04748 +0.04338 +0.03931 +0.03897 +0.04240 +0.04598 +0.04973 +0.05366 +0.05775 +0.06199 +0.06633 +0.07073 +0.07512 +0.07944 +0.08364 +0.08764 +0.09141 +0.09491 +0.09813 +0.10107 +0.10375 +0.06518 +0.06463 +0.06400 +0.06336 +0.06281 +0.06241 +0.06222 +0.06232 +0.06271 +0.06342 +0.06442 +0.06569 +0.06715 +0.06874 +0.07035 +0.07189 +0.07326 +0.07438 +0.07516 +0.07554 +0.07550 +0.07503 +0.07415 +0.07290 +0.07137 +0.06964 +0.06783 +0.06604 +0.06438 +0.06297 +0.06189 +0.06121 +0.06096 +0.06115 +0.06176 +0.06275 +0.06402 +0.06549 +0.06702 +0.06851 +0.06982 +0.07083 +0.07145 +0.07161 +0.07124 +0.07034 +0.06891 +0.06702 +0.06474 +0.06216 +0.05943 +0.05666 +0.05400 +0.05158 +0.04952 +0.04792 +0.04686 +0.04636 +0.04644 +0.04707 +0.04819 +0.04971 +0.05152 +0.05348 +0.05545 +0.05731 +0.05891 +0.06014 +0.06091 +0.06114 +0.06081 +0.05992 +0.05851 +0.05663 +0.05440 +0.05192 +0.04934 +0.04679 +0.04444 +0.04241 +0.04083 +0.03981 +0.03941 +0.03970 +0.04067 +0.04231 +0.04455 +0.04733 +0.05053 +0.05401 +0.05765 +0.06130 +0.06481 +0.06806 +0.07092 +0.07331 +0.07515 +0.07642 +0.07710 +0.07722 +0.07684 +0.07604 +0.07492 +0.07361 +0.07224 +0.07093 +0.06983 +0.06906 +0.06873 +0.06893 +0.06972 +0.07114 +0.07320 +0.07588 +0.07914 +0.08289 +0.08704 +0.09149 +0.09610 +0.10074 +0.10527 +0.10957 +0.11351 +0.11698 +0.11990 +0.12219 +0.12381 +0.12474 +0.12499 +0.12458 +0.12357 +0.12203 +0.12005 +0.11773 +0.11518 +0.11252 +0.10986 +0.10731 +0.10497 +0.10293 +0.10125 +0.09998 +0.09917 +0.09883

+0.09895 +0.09950 +0.10043 +0.10170 +0.10322 +0.10490 +0.10666 +0.10841 +0.11004 +0.11145 +0.11257 +0.11332 +0.11362 +0.11343 +0.11269 +0.11140 +0.10954 +0.10712
+0.10417 +0.10072 +0.09684 +0.09258 +0.08802 +0.08323 +0.07832 +0.07336 +0.06845 +0.06367 +0.05910 +0.05483 +0.05092 +0.04742 +0.04440 +0.04188 +0.03990 +0.03846
+0.03757 +0.03723 +0.03740 +0.03806 +0.03917 +0.04069 +0.04257 +0.04474 +0.04716 +0.04975 +0.05245 +0.05522 +0.05798 +0.06068 +0.06327 +0.06571 +0.06795 +0.06996
+0.07171 +0.07318 +0.07436 +0.07523 +0.07580 +0.07607 +0.07605 +0.07575 +0.07519 +0.07440 +0.07340 +0.07221 +0.07087 +0.06941 +0.06785 +0.06623 +0.06457 +0.06291
+0.06127 +0.05967 +0.05814 +0.05670 +0.05535 +0.05412 +0.05302 +0.05205 +0.05122 +0.05053 +0.04997 +0.04956 +0.04928 +0.04912 +0.04909 +0.04916 +0.04932 +0.04957
+0.04990 +0.05028 +0.05071 +0.05118 +0.05167 +0.05217 +0.05266 +0.05315 +0.05361 +0.05405 +0.05445 +0.05480 +0.05511 +0.05536 +0.05555 +0.05568 +0.05576 +0.05577
+0.05572 +0.05562 +0.05545 +0.05523 +0.05496 +0.05464 +0.05428 +0.05388 +0.05343 +0.05296 +0.05246 +0.05193 +0.05138 +0.05082 +0.05024 +0.04965 +0.04905 +0.04844
+0.04783 +0.04722 +0.04660 +0.04598 +0.04536 +0.04474 +0.04412 +0.04349 +0.06215 +0.06387 +0.06558 +0.06728 +0.06898 +0.07066 +0.07233 +0.07398 +0.07562 +0.07724
+0.07884 +0.08043 +0.08199 +0.08353 +0.08505 +0.08655 +0.08803 +0.08949 +0.08562 +0.08174 +0.07785 +0.07394 +0.07002 +0.06610 +0.06217 +0.05824 +0.05432 +0.05041
+0.04650 +0.04261 +0.03874 +0.03489 +0.03107 +0.02728 +0.02353 +0.01981 +0.01614 +0.03079 +0.03797 +0.04519 +0.05246 +0.05979 +0.09054 +0.09820 +0.10594 +0.11375
+0.12163 +0.12959 +0.13010 +0.12745 +0.12164 +0.10836 +0.09515 +0.08199 +0.07419 +0.06644 +0.05873 +0.05106 +0.04341 +0.03579 +0.02819 +0.02060 +0.01625 +0.01942
+0.02258 +0.02572 +0.02884 +0.03194 +0.03500 +0.03801 +0.04098 +0.04390 +0.04675 +0.07880 +0.07913 +0.07938 +0.07955 +0.07962 +0.07961 +0.07950 +0.07929 +0.07898
+0.07856 +0.07803 +0.07739 +0.07663 +0.07576 +0.07477 +0.07366 +0.07243 +0.07108 +0.06961 +0.06802 +0.06631 +0.06448 +0.06255 +0.06050 +0.05834 +0.05608 +0.05372
+0.05126 +0.04872 +0.04611 +0.04342 +0.04067 +0.03787 +0.03503 +0.03217 +0.02929 +0.02642 +0.02357 +0.02075 +0.01798 +0.01529 +0.01269 +0.01021 +0.00785 +0.00565
+0.00362 +0.00180 +0.00018 -0.00119 -0.00231 -0.00315 -0.00371 -0.00396 -0.00389 -0.00349 -0.00276 -0.00169 -0.00029 +0.00145 +0.00351 +0.00588 +0.00855 +0.01149 +0.01467
+0.01806 +0.02164 +0.02534 +0.02914 +0.03298 +0.03680 +0.04056 +0.04419 +0.04764 +0.05084 +0.05374 +0.05628 +0.05839 +0.06004 +0.06116 +0.06173 +0.06169 +0.06103
+0.05972 +0.05775 +0.05512 +0.05185 +0.04796 +0.04347 +0.03844 +0.03292 +0.02697 +0.02069 +0.01414 +0.00743 +0.00066 -0.00607 -0.01264 -0.01896 -0.02490 -0.03038 -
0.03528 -0.03953 -0.04303 -0.04573 -0.04758 -0.04854 -0.04859 -0.04774 -0.04601 -0.04344 -0.04010 -0.03606 -0.03142 -0.02629 -0.02081 -0.01509 -0.00929 -0.00354 +0.00200
+0.00719 +0.01192 +0.01606 +0.01950 +0.02216 +0.02399 +0.02495 +0.02502 +0.02424 +0.02263 +0.02028 +0.01728 +0.01376 +0.00984 +0.00569 +0.00146 -0.00266 -0.00652 -
0.00995 -0.01279 -0.01492 -0.01623 -0.01663 -0.01607 -0.01453 -0.01203 -0.00862 -0.00439 +0.00054 +0.00601 +0.01186 +0.01789 +0.02389 +0.02966 +0.03498 +0.03966 +0.04353
+0.04642 +0.04823 +0.04886 +0.04828 +0.04650 +0.04355 +0.03954 +0.03460 +0.02889 +0.02262 +0.01600 +0.00927 +0.00266 -0.00358 -0.00926 -0.01417 -0.01817 -0.02114 -
0.02301 -0.02376 -0.02344 -0.02212 -0.01994 -0.01706 -0.01371 -0.01010 -0.00648 -0.00310 -0.00020 +0.00202 +0.00336 +0.00368 +0.00289 +0.00096 -0.00210 -0.00622 -0.01127

-0.01707 -0.02341 -0.03004 -0.03670 -0.04313 -0.04906 -0.05428 -0.05857 -0.06178 -0.06381 -0.06462 -0.06424 -0.06273 -0.06024 -0.05694 -0.05307 -0.04888 -0.04463 -0.04058 -0.03698 -0.03406 -0.03198 -0.03089 -0.03083 -0.03182 -0.03380 -0.03664 -0.04019 -0.04423 -0.04851 -0.05278 -0.05677 -0.06024 -0.06297 -0.06478 -0.06556 -0.06524 -0.06382 -0.06135 -0.05797 -0.05384 -0.04918 -0.04422 -0.03922 -0.03444 -0.03010 -0.02640 -0.02351 -0.02153 -0.02050 -0.02040 -0.02115 -0.02263 -0.02466 -0.02702 -0.02948 -0.03182 -0.03380 -0.03523 -0.03595 -0.03585 -0.03486 -0.03299 -0.03030 -0.02688 -0.02291 -0.01855 -0.01404 -0.00957 -0.00535 -0.00158 +0.00160 +0.00407 +0.00578 +0.00671 +0.00692 +0.00649 +0.00557 +0.00430 +0.00287 +0.00147 +0.00028 -0.00054 -0.00087 -0.00061 +0.00028 +0.00180 +0.00391 +0.00651 +0.00950 +0.01273 +0.01605 -0.04774 -0.04399 -0.04057 -0.03760 -0.03513 -0.03319 -0.03176 -0.03080 -0.03022 -0.02994 -0.02982 -0.02974 -0.02959 -0.02926 -0.02866 -0.02773 -0.02644 -0.02478 -0.02069 -0.01632 -0.01172 -0.00700 -0.00223 +0.00249 +0.00709 +0.01150 +0.01568 +0.01960 +0.02327 +0.02671 +0.02994 +0.03303 +0.03602 +0.03897 +0.04195 +0.04500 +0.04814 +0.04904 +0.02880 +0.00866 -0.01139 -0.01013 -0.00649 +0.00893 +0.03610 -0.01136 -0.00078 +0.00969 +0.01456 +0.01698 +0.01938 +0.02178 +0.02423 +0.02678 +0.02736 +0.02809 +0.02899 +0.03003 +0.03119 +0.03244 +0.03370 +0.03492 +0.03603 +0.03695 +0.03764 +0.03805 +0.03816 +0.03797 +0.03752 +0.03684 +0.03602 +0.03515 +0.03431 -0.06649 -0.06763 -0.06844 -0.06887 -0.06887 -0.06844 -0.06762 -0.06646 -0.06506 -0.06352 -0.06198 -0.06057 -0.05942 -0.05864 -0.05834 -0.05857 -0.05936 -0.06069 -0.06249 -0.06468 -0.06713 -0.06968 -0.07216 -0.07441 -0.07627 -0.07760 -0.07830 -0.07831 -0.07759 -0.07619 -0.07419 -0.07170 -0.06888 -0.06591 -0.06300 -0.06032 -0.05807 -0.05640 -0.05542 -0.05520 -0.05577 -0.05706 -0.05901 -0.06146 -0.06424 -0.06714 -0.06994 -0.07241 -0.07435 -0.07558 -0.07597 -0.07542 -0.07392 -0.07147 -0.06819 -0.06420 -0.05970 -0.05490 -0.05004 -0.04537 -0.04111 -0.03748 -0.03463 -0.03270 -0.03173 -0.03172 -0.03262 -0.03429 -0.03656 -0.03922 -0.04202 -0.04472 -0.04706 -0.04880 -0.04975 -0.04975 -0.04869 -0.04653 -0.04330 -0.03908 -0.03401 -0.02829 -0.02215 -0.01583 -0.00962 -0.00378 +0.00145 +0.00586 +0.00927 +0.01157 +0.01270 +0.01268 +0.01156 +0.00948 +0.00661 +0.00317 -0.00060 -0.00443 -0.00808 -0.01128 -0.01382 -0.01551 -0.01624 -0.01590 -0.01449 -0.01204 -0.00866 -0.00448 +0.00029 +0.00544 +0.01070 +0.01582 +0.02054 +0.02462 +0.02783 +0.03000 +0.03099 +0.03072 +0.02917 +0.02635 +0.02235 +0.01730 +0.01137 +0.00477 -0.00227 -0.00950 -0.01668 -0.02356 -0.02992 -0.03557 -0.04035 -0.04414 -0.04688 -0.04855 -0.04917 -0.04882 -0.04761 -0.04569 -0.04325 -0.04049 -0.03762 -0.03486 -0.03242 -0.03050 -0.02927 -0.02887 -0.02941 -0.03097 -0.03357 -0.03721 -0.04182 -0.04732 -0.05359 -0.06048 -0.06779 -0.07536 -0.08296 -0.09040 -0.09747 -0.10399 -0.10979 -0.11472 -0.11866 -0.12151 -0.12323 -0.12378 -0.12317 -0.12146 -0.11870 -0.11500 -0.11048 -0.10528 -0.09956 -0.09348 -0.08722 -0.08093 -0.07477 -0.06891 -0.06347 -0.05858 -0.05433 -0.05080 -0.04805 -0.04610 -0.04495 -0.04459 -0.04499 -0.04608 -0.04779 -0.05002 -0.05269 -0.05567 -0.05886 -0.06213 -0.06539 -0.06850 -0.07139 -0.07394 -0.07608 -0.07775 -0.07888 -0.07943 -0.07939 -0.07875 -0.07750 -0.07567 -0.07330 -0.07043 -0.06712 -0.06344 -0.05945 -0.05524 -0.05089 -0.04649 -0.04211 -0.03785 -0.03378 -0.02998 -0.02651 -0.02345 -0.02083 -0.01872 -0.01715 -0.01615 -0.01573 -0.01590 -0.01666 -0.01801 -0.01993 -0.02239 -0.02535 -0.02879 -0.03266 -0.03690 -0.04146 -0.04630 -0.05135 -0.05655 -0.06185 -0.06719 -0.07251 -0.07775 -0.08287 -0.08782 -0.09254 -0.09700 -0.10115 -0.10498 -0.10844 -

0.11151 -0.11418 -0.11643 -0.11825 -0.11963 -0.12057 -0.12108 -0.12116 -0.12082 -0.12008 -0.11895 -0.11746 -0.11561 -0.11345 -0.11098 -0.10824 -0.10526 -0.10206 -0.09867 -0.09512 -0.09144 -0.08765 -0.08379 -0.07987 -0.07594 -0.07200 -0.06808 -0.06421 -0.06040 -0.05667 -0.05304 -0.04952 -0.04613 -0.04288 -0.03977 -0.03681 -0.03401 -0.03138 -0.02891 -0.02660 -0.02446 -0.02249 -0.02067 -0.01901 -0.01750 -0.01614 -0.01491 -0.01380 -0.01282 -0.01194 -0.01116 -0.01048 -0.00987 -0.00933 -0.00885 -0.01246 -0.01352 -0.01461 -0.01572 -0.01685 -0.01798 -0.01911 -0.02024 -0.02134 -0.02243 -0.02350 -0.02454 -0.02555 -0.02653 -0.02747 -0.02838 -0.02927 -0.03011 -0.03093 -0.03173 -0.03250 -0.03324 -0.03397 -0.03469 -0.03540 -0.03611 -0.03682 -0.03753</SIMULATION>

</ANALYSIS>

<ANALYSIS NAME="ANSI_B_5_54">

<NUMBER_OF_ERRORS>0</NUMBER_OF_ERRORS>

<NUMBER_OF_WARNINGS>0</NUMBER_OF_WARNINGS>

<FEATURE NAME="AF_NON_ROUNDNESS" DT="UT_LENGTH_UM">439.5</FEATURE>

<FEATURE NAME="AF_MAX_CIRCULAR_DEVIATION" DT="UT_LENGTH_UM">218.4</FEATURE>

<FEATURE NAME="AF_MAX_CIRCULAR_DEVIATION_LOCATION" DT="UT_ANGLE">180.79245</FEATURE>

<FEATURE NAME="AF_MIN_CIRCULAR_DEVIATION" DT="UT_LENGTH_UM">-221.1</FEATURE>

<FEATURE NAME="AF_MIN_CIRCULAR_DEVIATION_LOCATION" DT="UT_ANGLE">216.57420</FEATURE>

<FEATURE NAME="AF_BEST_RADIUS" DT="UT_LENGTH_MM">100.0002</FEATURE>

<FEATURE NAME="AF_CENTRE_OFFSET_A" DT="UT_LENGTH_UM">-64.5</FEATURE>

<FEATURE NAME="AF_CENTRE_OFFSET_B" DT="UT_LENGTH_UM">-18.9</FEATURE>

<FEATURE NAME="AF_CALCULATED_FEEDRATE" DT="UT_FEEDRATE">4985.4</FEATURE>

</ANALYSIS>

<ANALYSIS NAME="ANSI_B_5_54" version="2005">

<FEATURE NAME="AF_CIRCULAR_DEVIATION_CW" DT="UT_LENGTH_UM">424.4</FEATURE>

<FEATURE NAME="AF_MAX_CIRCULAR_DEVIATION_CW" DT="UT_LENGTH_UM">197.1</FEATURE>

<FEATURE NAME="AF_MIN_CIRCULAR_DEVIATION_CW" DT="UT_LENGTH_UM">-227.4</FEATURE>

<FEATURE NAME="AF_MAX_CIRCULAR_DEVIATION_LOCATION_CW" DT="UT_ANGLE">8.61304</FEATURE>

<FEATURE NAME="AF_MIN_CIRCULAR_DEVIATION_LOCATION_CW" DT="UT_ANGLE">216.57420</FEATURE>

<FEATURE NAME="AF_RADIAL_DEVIATION_CW" DT="UT_LENGTH_UM">424.4</FEATURE>

<FEATURE NAME="AF_MAX_RADIAL_DEVIATION_CW" DT="UT_LENGTH_UM">202.2</FEATURE>

<FEATURE NAME="AF_MAX_RADIAL_DEVIATION_LOCATION_CW" DT="UT_ANGLE">8.61304</FEATURE>

<FEATURE NAME="AF_MIN_RADIAL_DEVIATION_CW" DT="UT_LENGTH_UM">-222.3</FEATURE>

<FEATURE NAME="AF_MIN_RADIAL_DEVIATION_LOCATION_CW" DT="UT_ANGLE">216.57420</FEATURE>

<FEATURE NAME="AF_CIRCULAR_DEVIATION_CCW" DT="UT_LENGTH_UM">393.3</FEATURE>

<FEATURE NAME="AF_MAX_CIRCULAR_DEVIATION_CCW" DT="UT_LENGTH_UM">210.1</FEATURE>

<FEATURE NAME="AF_MIN_CIRCULAR_DEVIATION_CCW" DT="UT_LENGTH_UM">-183.2</FEATURE>

<FEATURE NAME="AF_MAX_CIRCULAR_DEVIATION_LOCATION_CCW" DT="UT_ANGLE">180.79245</FEATURE>

<FEATURE NAME="AF_MIN_CIRCULAR_DEVIATION_LOCATION_CCW" DT="UT_ANGLE">173.32075</FEATURE>

<FEATURE NAME="AF_RADIAL_DEVIATION_CCW" DT="UT_LENGTH_UM">393.3</FEATURE>

<FEATURE NAME="AF_MAX_RADIAL_DEVIATION_CCW" DT="UT_LENGTH_UM">205.5</FEATURE>

<FEATURE NAME="AF_MAX_RADIAL_DEVIATION_LOCATION_CCW" DT="UT_ANGLE">180.79245</FEATURE>

<FEATURE NAME="AF_MIN_RADIAL_DEVIATION_CCW" DT="UT_LENGTH_UM">-187.8</FEATURE>

<FEATURE NAME="AF_MIN_RADIAL_DEVIATION_LOCATION_CCW" DT="UT_ANGLE">173.32075</FEATURE>

<FEATURE NAME="AF_CIRCULAR_DEVIATION_BIDIRECTIONAL" DT="UT_LENGTH_UM">437.4</FEATURE>

</ANALYSIS>

<ANALYSIS NAME="ANSI_B_5_57">

<FEATURE NAME="AF_CIRCULAR_DEVIATION_CW" DT="UT_LENGTH_UM">424.4</FEATURE>

<FEATURE NAME="AF_MAX_CIRCULAR_DEVIATION_CW" DT="UT_LENGTH_UM">197.1</FEATURE>

<FEATURE NAME="AF_MIN_CIRCULAR_DEVIATION_CW" DT="UT_LENGTH_UM">-227.4</FEATURE>

<FEATURE NAME="AF_MAX_CIRCULAR_DEVIATION_LOCATION_CW" DT="UT_ANGLE">8.61304</FEATURE>

<FEATURE NAME="AF_MIN_CIRCULAR_DEVIATION_LOCATION_CW" DT="UT_ANGLE">216.57420</FEATURE>

<FEATURE NAME="AF_RADIAL_DEVIATION_CW" DT="UT_LENGTH_UM">424.4</FEATURE>

<FEATURE NAME="AF_MAX_RADIAL_DEVIATION_CW" DT="UT_LENGTH_UM">202.2</FEATURE>

<FEATURE NAME="AF_MAX_RADIAL_DEVIATION_LOCATION_CW" DT="UT_ANGLE">8.61304</FEATURE>

<FEATURE NAME="AF_MIN_RADIAL_DEVIATION_CW" DT="UT_LENGTH_UM">-222.3</FEATURE>

<FEATURE NAME="AF_MIN_RADIAL_DEVIATION_LOCATION_CW" DT="UT_ANGLE">216.57420</FEATURE>

<FEATURE NAME="AF_CIRCULAR_DEVIATION_CCW" DT="UT_LENGTH_UM">393.3</FEATURE>

<FEATURE NAME="AF_MAX_CIRCULAR_DEVIATION_CCW" DT="UT_LENGTH_UM">210.1</FEATURE>

<FEATURE NAME="AF_MIN_CIRCULAR_DEVIATION_CCW" DT="UT_LENGTH_UM">-183.2</FEATURE>

<FEATURE NAME="AF_MAX_CIRCULAR_DEVIATION_LOCATION_CCW" DT="UT_ANGLE">180.79245</FEATURE>

<FEATURE NAME="AF_MIN_CIRCULAR_DEVIATION_LOCATION_CCW" DT="UT_ANGLE">173.32075</FEATURE>

<FEATURE NAME="AF_RADIAL_DEVIATION_CCW" DT="UT_LENGTH_UM">393.3</FEATURE>

<FEATURE NAME="AF_MAX_RADIAL_DEVIATION_CCW" DT="UT_LENGTH_UM">205.5</FEATURE>

<FEATURE NAME="AF_MAX_RADIAL_DEVIATION_LOCATION_CCW" DT="UT_ANGLE">180.79245</FEATURE>

<FEATURE NAME="AF_MIN_RADIAL_DEVIATION_CCW" DT="UT_LENGTH_UM">-187.8</FEATURE>

<FEATURE NAME="AF_MIN_RADIAL_DEVIATION_LOCATION_CCW" DT="UT_ANGLE">173.32075</FEATURE>

<FEATURE NAME="AF_CIRCULAR_DEVIATION_BIDIRECTIONAL" DT="UT_LENGTH_UM">437.4</FEATURE>

</ANALYSIS>

<ANALYSIS NAME="ISO230_4" version="1996">

<FEATURE NAME="AF_CIRCULAR_DEVIATION_CCW" DT="UT_LENGTH_UM">393.3</FEATURE>

<FEATURE NAME="AF_MAX_CIRCULAR_DEVIATION_CCW" DT="UT_LENGTH_UM">210.1</FEATURE>

<FEATURE NAME="AF_MIN_CIRCULAR_DEVIATION_CCW" DT="UT_LENGTH_UM">-183.2</FEATURE>

<FEATURE NAME="AF_MAX_CIRCULAR_DEVIATION_LOCATION_CCW" DT="UT_ANGLE">180.79245</FEATURE>

<FEATURE NAME="AF_MIN_CIRCULAR_DEVIATION_LOCATION_CCW" DT="UT_ANGLE">173.32075</FEATURE>

<FEATURE NAME="AF_RADIAL_DEVIATION_CCW" DT="UT_LENGTH_UM">393.3</FEATURE>

<FEATURE NAME="AF_MAX_RADIAL_DEVIATION_CCW" DT="UT_LENGTH_UM">205.5</FEATURE>

<FEATURE NAME="AF_MIN_RADIAL_DEVIATION_CCW" DT="UT_LENGTH_UM">-187.8</FEATURE>

<FEATURE NAME="AF_MAX_RADIAL_DEVIATION_LOCATION_CCW" DT="UT_ANGLE">180.79245</FEATURE>

<FEATURE NAME="AF_MIN_RADIAL_DEVIATION_LOCATION_CCW" DT="UT_ANGLE">173.32075</FEATURE>

<FEATURE NAME="AF_CIRCULAR_DEVIATION_CW" DT="UT_LENGTH_UM">424.4</FEATURE>

<FEATURE NAME="AF_MAX_CIRCULAR_DEVIATION_CW" DT="UT_LENGTH_UM">197.1</FEATURE>

<FEATURE NAME="AF_MIN_CIRCULAR_DEVIATION_CW" DT="UT_LENGTH_UM">-227.4</FEATURE>

<FEATURE NAME="AF_MAX_CIRCULAR_DEVIATION_LOCATION_CW" DT="UT_ANGLE">8.61304</FEATURE>

<FEATURE NAME="AF_MIN_CIRCULAR_DEVIATION_LOCATION_CW" DT="UT_ANGLE">216.57420</FEATURE>

<FEATURE NAME="AF_RADIAL_DEVIATION_CW" DT="UT_LENGTH_UM">424.4</FEATURE>

<FEATURE NAME="AF_MAX_RADIAL_DEVIATION_CW" DT="UT_LENGTH_UM">202.2</FEATURE>

<FEATURE NAME="AF_MIN_RADIAL_DEVIATION_CW" DT="UT_LENGTH_UM">-222.3</FEATURE>

<FEATURE NAME="AF_MAX_RADIAL_DEVIATION_LOCATION_CW" DT="UT_ANGLE">8.61304</FEATURE>

<FEATURE NAME="AF_MIN_RADIAL_DEVIATION_LOCATION_CW" DT="UT_ANGLE">216.57420</FEATURE>

<FEATURE NAME="AF_CIRCULAR_DEVIATION_BIDIRECTIONAL" DT="UT_LENGTH_UM">437.4</FEATURE>

<FEATURE NAME="AF_CIRCULAR_HYSTERESIS" DT="UT_LENGTH_UM">313.7</FEATURE>

<FEATURE NAME="AF_CIRCULAR_HYSTERESIS_LOCATION" DT="UT_ANGLE">199.34813</FEATURE>

</ANALYSIS>

<ANALYSIS NAME="ISO230_4" version="2005">

<FEATURE NAME="AF_CIRCULAR_DEVIATION_CCW" DT="UT_LENGTH_UM">393.3</FEATURE>

<FEATURE NAME="AF_MAX_CIRCULAR_DEVIATION_CCW" DT="UT_LENGTH_UM">210.1</FEATURE>

<FEATURE NAME="AF_MIN_CIRCULAR_DEVIATION_CCW" DT="UT_LENGTH_UM">-183.2</FEATURE>

<FEATURE NAME="AF_MAX_CIRCULAR_DEVIATION_LOCATION_CCW" DT="UT_ANGLE">180.79245</FEATURE>

<FEATURE NAME="AF_MIN_CIRCULAR_DEVIATION_LOCATION_CCW" DT="UT_ANGLE">173.32075</FEATURE>

<FEATURE NAME="AF_RADIAL_DEVIATION_CCW" DT="UT_LENGTH_UM">393.3</FEATURE>

<FEATURE NAME="AF_MAX_RADIAL_DEVIATION_CCW" DT="UT_LENGTH_UM">205.5</FEATURE>

<FEATURE NAME="AF_MIN_RADIAL_DEVIATION_CCW" DT="UT_LENGTH_UM">-187.8</FEATURE>

<FEATURE NAME="AF_MAX_RADIAL_DEVIATION_LOCATION_CCW" DT="UT_ANGLE">180.79245</FEATURE>

<FEATURE NAME="AF_MIN_RADIAL_DEVIATION_LOCATION_CCW" DT="UT_ANGLE">173.32075</FEATURE>

<FEATURE NAME="AF_CIRCULAR_DEVIATION_CW" DT="UT_LENGTH_UM">424.4</FEATURE>

<FEATURE NAME="AF_MAX_CIRCULAR_DEVIATION_CW" DT="UT_LENGTH_UM">197.1</FEATURE>

<FEATURE NAME="AF_MIN_CIRCULAR_DEVIATION_CW" DT="UT_LENGTH_UM">-227.4</FEATURE>

<FEATURE NAME="AF_MAX_CIRCULAR_DEVIATION_LOCATION_CW" DT="UT_ANGLE">8.61304</FEATURE>

<FEATURE NAME="AF_MIN_CIRCULAR_DEVIATION_LOCATION_CW" DT="UT_ANGLE">216.57420</FEATURE>

<FEATURE NAME="AF_RADIAL_DEVIATION_CW" DT="UT_LENGTH_UM">424.4</FEATURE>

<FEATURE NAME="AF_MAX_RADIAL_DEVIATION_CW" DT="UT_LENGTH_UM">202.2</FEATURE>

<FEATURE NAME="AF_MIN_RADIAL_DEVIATION_CW" DT="UT_LENGTH_UM">-222.3</FEATURE>

<FEATURE NAME="AF_MAX_RADIAL_DEVIATION_LOCATION_CW" DT="UT_ANGLE">8.61304</FEATURE>

<FEATURE NAME="AF_MIN_RADIAL_DEVIATION_LOCATION_CW" DT="UT_ANGLE">216.57420</FEATURE>

<FEATURE NAME="AF_BIDIRECTIONAL_CIRCULAR_DEVIATION" DT="UT_LENGTH_UM">439.5</FEATURE>

<FEATURE NAME="AF_MEAN_BIDIRECTIONAL_RADIAL_DEVIATION" DT="UT_LENGTH_UM">0.3</FEATURE>

</ANALYSIS>

</TEST_DOCUMENT>

16.4 COBOT code to collect data

```
%% new combined code for controling the COBOT and adding the displacment readings
clc
clear
close all

%% Notes
% 83.3 = 5000 mm/min
% 100 = 6000 mm/min
%% COBOT set up code
RDK = Robolink;
%global Varibles and settings for arm
SimSpeed = 1; %vary from 0 to 10
ArmSpeed = 83.3; %sets speed to mm/s
ArmAccel = 83.3; %set accel to mm/s2
MaterialThick = 5; %thickness of material in mm
MPose = 0; %location of Materal realtive to base

%assigns verables of the RoboDK items to veriables
%Ball Bar Test
BallBarTestXY = RDK.Item('Ball Bar Test XY');
BallBarTestYZ = RDK.Item('Ball Bar YZ ');
BallBarTestXZ = RDK.Item('Ball Bar XZ');
CalibrationTest = RDK.Item('Calibration Test');
ListProg = [BallBarTestXY];     %Z axis drifts
%ListProg = [BallBarTestYZ];     %X axis drifts
```

```
%ListProg = [BallBarTestXZ];     %Y axis drifts
%ListProg =[CalibrationTest];

%==== Parts ====
robot = RDK.Item('UR10e');%when you know other wise use line below
ToolTCP = RDK.Item('COBOT Ball Bar Mount with Tri'); %get the TCP of tool %RobotiQ HandE Gripper
ToolTCP = RDK.Item('RobotiQ Hand-E Gripper'); %get the TCP of tool
TAcell = RDK.Item('Uni Tri Mount');

%==== Pose varibles =====
count = 0;
XYZPose = [];%poseition of robot wrist
XYZToolTCP = [];%poseition of tool tcp
XYZTCPtoBase = [];%TCP in ref to base
OFFXYZPose = [];

differenceInPose =[1,0,0,-0.0006;
                   0,1,0,-58.4987;
                   0,0,1,-66.2986;
                   0,0,0,1];

%% setting up NI and Accel reading
%connecting to board
accel = daq("ni");
%|Accel 1 for Arm/CNC|Accel for Base|Vibration|
C0 = addinput(accel, "cDAQ1Mod1", 0, "Accelerometer");%X
C1 = addinput(accel, "cDAQ1Mod1", 1, "Accelerometer");%Y
C2 = addinput(accel, "cDAQ1Mod1", 2, "Accelerometer");%Z
C0.Sensitivity = 1;
C1.Sensitivity = 1;
C2.Sensitivity = 1;
```

```
B0 = addinput(accel, "cDAQ1Mod2", 0, "Accelerometer");%X
B1 = addinput(accel, "cDAQ1Mod2", 1, "Accelerometer");%Y
B2 = addinput(accel, "cDAQ1Mod2", 2, "Accelerometer");%Z
B0.Sensitivity = 1;
B1.Sensitivity = 1;
B2.Sensitivity = 1;
NumScan = 800                              %number of scans
%disp(accel)

t = 0.00009765625;
SR = 10240;                                  %Sample rate = 10240|5120
SF = 1;                                      % customised units scale factor, units/volt
accel.Rate = SR;                             %sets samperling rate
count = 1/SR;                                %used as timer count
CorrectedXYZ =[];
OffCorrectedXYZ =[];
AllAccelVales = [];
OffsetVoltageData =[];
OffsetVoltageDataBase =[];
AllXValues = [];
AllYValues = [];
AllZValues = [];
AllFFTXvalues =[];
AllFFTYvalues =[];
AllFFTZvalues =[];
AllCalAccelTCP = [];

%% ==== running the RoboDK Code  ====
RDK.setSimulationSpeed(SimSpeed);
start(accel,"Continuous")
pause(1)
```

```
disp('starting offset')
%RawOffsetData = read(accel,seconds(5));     %collects same amount of scans to removed from attachal data
for CollectOffSetData = 0: 1: 5
    RawOffsetData = read(accel,"NumScans",NumScan);
    CorrectedOffsetData = [RawOffsetData.cDAQ1Mod1_ai0,RawOffsetData.cDAQ1Mod1_ai1,RawOffsetData.cDAQ1Mod1_ai2];
                                                                CorrectedOffsetDataBase                                =
[RawOffsetData.cDAQ1Mod2_ai0,RawOffsetData.cDAQ1Mod2_ai1,RawOffsetData.cDAQ1Mod2_ai2];
     OffsetVoltageData = [OffsetVoltageData;CorrectedOffsetData];                %stores data in a uasable and editable
formate with time
    OffsetVoltageDataBase = [OffsetVoltageDataBase;CorrectedOffsetDataBase];               %stores data in a uasable and
editable formate with time
    pose = robot.Pose();
    CurrentXYZ = [ pose(1,4) pose(2,4) pose(3,4)]; %current Pose cartiesa
    OFFXYZPose = [OFFXYZPose;CurrentXYZ];
    pause(1)
end
MeanOffset = mean(OffsetVoltageData)
MeanOffsetBase = mean(OffsetVoltageDataBase)%acts a datum for aceel values
MeanOffSetPose = mean(OFFXYZPose);% so I have one piont to measure to
disp('offset done');

%% ==== runs Ball bar test
for n = 1 : length(ListProg)
    ListProg(n).RunProgram();
    while ListProg(n).Busy()
        %Pos values from COBOT
        pose = robot.Pose();%seems to give wrist TCP not Tool TCP
        CurrentXYZ = [ pose(1,4) pose(2,4) pose(3,4)]; %current Pose cartiesa
        XYZPose = [XYZPose;CurrentXYZ];

        %Displacment values from Accel
```

```
        ReadRawData = read(accel,"NumScans",NumScan); %need to use read, also need to do more then one scan
                                                                                    RawScanData                    =
[seconds(ReadRawData.Time),ReadRawData.cDAQ1Mod1_ai0,ReadRawData.cDAQ1Mod1_ai1,ReadRawData.cDAQ1Mod1_ai2];
                                                                                RawScanDataBase                    =
[seconds(ReadRawData.Time),ReadRawData.cDAQ1Mod2_ai0,ReadRawData.cDAQ1Mod2_ai1,ReadRawData.cDAQ1Mod2_ai2];

        %FFT
        CutOff =0.0006; %cut off for power septrume
        Len = NumScan;

        %X FFT
        fhatX = fft(RawScanData(:,2),Len);
        PSDX = fhatX.*conj(fhatX)/Len;
        freq = 1/(t(1)*Len)*(0:Len);
        LPlotX = 1:floor(Len/2);
        indicesX = PSDX>CutOff;
        PSDCleanX = PSDX.*indicesX;
        fhatX = indicesX.*fhatX;
        ffiltX = ifft(fhatX);
        %Y FFT
        fhatY = fft(RawScanData(:,3),Len);
        PSDY = fhatY.*conj(fhatY)/Len;
        LPlotY = 1:floor(Len/2);
        indicesY = PSDY>CutOff;
        PSDCleanY = PSDY.*indicesY;
        fhatY = indicesY.*fhatY;
        ffiltY = ifft(fhatY);
        %Z FFT
        fhatZ = fft(RawScanData(:,4),Len);
        PSDZ = fhatZ.*conj(fhatZ)/Len;
        LPlotZ = 1:floor(Len/2);
```

```
indicesZ = PSDZ>CutOff;
PSDCleanZ = PSDZ.*indicesZ;
fhatZ = indicesZ.*fhatZ;
ffiltZ = ifft(fhatZ);

%X FFT Base
fhatXBase = fft(RawScanDataBase(:,2),Len);
PSDXBase = fhatXBase.*conj(fhatXBase)/Len;
freq = 1/(t(1)*Len)*(0:Len);                %I think I can comment this line out
LPlotXBase = 1:floor(Len/2);                %I think I can comment this line out
indicesXBase = PSDXBase>CutOff;
PSDCleanXBase = PSDXBase.*indicesXBase;     %I think I can comment this line out
fhatXBase = indicesXBase.*fhatXBase;
ffiltXBase = ifft(fhatXBase);
%Y FFT Base
fhatYBase = fft(RawScanDataBase(:,3),Len);
PSDYBase = fhatYBase.*conj(fhatYBase)/Len;
freq = 1/(t(1)*Len)*(0:Len);
LPlotYBase = 1:floor(Len/2);
indicesYBase = PSDYBase>CutOff;             %I think I can comment this line out
PSDCleanYBase = PSDYBase.*indicesYBase;     %I think I can comment this line out
fhatYBase = indicesYBase.*fhatYBase;
ffiltYBase = ifft(fhatYBase);
%Z FFT Base
fhatZBase = fft(RawScanDataBase(:,4),Len);
PSDZBase = fhatZBase.*conj(fhatZBase)/Len;
freq = 1/(t(1)*Len)*(0:Len);
LPlotZBase = 1:floor(Len/2);
indicesZBase = PSDZBase>CutOff;             %I think I can comment this line out
PSDCleanZBase = PSDZBase.*indicesZBase;     %I think I can comment this line out
fhatZBase = indicesZBase.*fhatZBase;
```

```
        ffiltZBase = ifft(fhatZBase);


        %collect all values
        AllXValues = [AllXValues;RawScanData(:,2)];
        AllFFTXvalues = [AllFFTXvalues;ffiltX];
        AllYValues = [AllYValues;RawScanData(:,3)];
        AllFFTYvalues = [AllFFTYvalues;ffiltY];
        AllZValues = [AllZValues;RawScanData(:,4)];
        AllFFTZvalues = [AllFFTZvalues;ffiltZ];

        % converts from G to mm/S^2 and applies offset
        %FFT and Offset
        Chan1V = ((ffiltX-MeanOffset(1)) *9806.65); %X
        Chan2V = ((ffiltY-MeanOffset(2)) *9806.65); %Y
        Chan3V = ((ffiltZ-MeanOffset(3)) *9806.65); %Z
        AccelData = [Chan1V,Chan2V,Chan3V];

        %double intergrration
        %X
        velocityX = (cumtrapz(t,Chan1V,1));
        DisplacementX= (cumtrapz(t,velocityX,1));
        %Y
        velocityY = (cumtrapz(t,Chan2V,1));
        DisplacementY= (cumtrapz(t,velocityY,1));
        %Z
        velocityZ = (cumtrapz(t,Chan3V,1));
        DisplacementZ= (cumtrapz(t,velocityZ,1));
        %All Dis Data
        DisData = [DisplacementX,DisplacementY,DisplacementZ];
        SumDisData = mean(DisData);%?
```

```
        AllAccelVales = [AllAccelVales;SumDisData];

        %tranformation cal from accel to Mag
        TTCP = ToolTCP.PoseAbs(); %gets robot wrist relative to world frame
        TAcellTCP = TAcell.PoseTool();%gets TCP acell relative to wrist
        CalATCP = TTCP*TAcellTCP;
        AllCalAccelTCP = [AllCalAccelTCP;CalATCP(1,4),CalATCP(2,4),CalATCP(3,4)];
        TranformToMagTCP = differenceInPose*CalATCP; %cal culates the Mag Stick TCP based of Accel Pose
        %550 and 185 is to reset the origin at the ball bar center, this is in ref to world frame not BallbarFrame
        TranformToMagTCP = [550 - TranformToMagTCP(1,4), TranformToMagTCP(2,4), 185 - TranformToMagTCP(3,4)];

        %line below was for calibrating to 0
        %TranformToMagTCP = [600 - TranformToMagTCP(1,4), TranformToMagTCP(2,4), TranformToMagTCP(3,4)];

        %line about for test changes it to a 3 part array
                        SumDisData     =     [SumDisData(1)+TranformToMagTCP(1),     SumDisData(2)+TranformToMagTCP(2),
SumDisData(3)+TranformToMagTCP(3)];
        %this was SumDisData is the XYZ data and is fully indapendent from
        %TrueXYZ
        %combine Dis and Pos
        TrueXYZ = [CurrentXYZ(1) + SumDisData(1),CurrentXYZ(2) + SumDisData(2),CurrentXYZ(3) + SumDisData(3)];
        %CorrectedXYZ = [CorrectedXYZ;TrueXYZ];
        CorrectedXYZ = [CorrectedXYZ;SumDisData];
        count = count +1;
    end
end
MeanCorrectedXYZ = mean(CorrectedXYZ);
%% ==displaying FFT
% plot (freq(LPlotX),PSDX(LPlotX),'r') %this gets the how ofter a frequency is so random ones can be removed
% title('Power spectrum X')
% ylabel({'Frequency','(Hz)'})
```

```
% xlabel('Amount')

%% ==== displaying data ====
fprintf('Done!\n')
pointsize = 5;  %adjust at will
scatter3(XYZPose(:,1), XYZPose(:,2), XYZPose(:,3),'g','*');%plots the robot wrist
hold on
scatter3(CorrectedXYZ(:,1),CorrectedXYZ(:,2),CorrectedXYZ(:,3),'blue','o')
pointsize = 20;  %adjust at will
hold off
grid on

%plot grapth of data
title('Arm TCP');
xlabel('X');
ylabel('Y');
zlabel('Z');
legend('Tool TCP')
```

16.5 Code to process COBOT data

```
clc

%need to make code that can read get the radius from XYZ and Ball bar

BBFile2Open = ("XY 360deg 300mm 6000 mmMin Calibrated Test 2 20230413-145357.b5r");  %change file name

MatData2Open = ("300 mm 6000mmMin Test 1.mat")

Radius = 300;   %change value to test R



CCWResults = [];

CWResults = [];

CCW_X = [];

CCW_Y = [];

CW_X = [];

CW_Y = [];
```

```
%loads data

Data = readstruct(BBFile2Open,"FileType","xml");

load(MatData2Open)

%gets all of the data into one strut

Results = Data.RUN_RESULTS.BALLBAR_RUN;



%splites the results in to a CCW and CW

CCW = struct2table(Results(1));

CW = struct2table(Results(2));



%collects the relavent data in a readable format
```

```
AllCCWResults = str2num(CCW.READINGS);

CCWStart_Index = CCW.START_INDEX;

CCWEnd_Index = CCW.END_INDEX;

AllCWResults = str2num(CW.READINGS);

CWStart_Index = CW.START_INDEX;

CWEnd_Index = CW.END_INDEX;



%Gets radius data pionts in correct radius

for Index = CCWStart_Index : CCWEnd_Index

    CCWResults = [CCWResults; AllCCWResults(Index)+Radius];

end

for Index = CWStart_Index : CWEnd_Index

    CWResults = [CWResults; AllCWResults(Index)+Radius];
```

```
end



%gets valid data pionts and means them

CCWMeanIndexValue = round(length(CCWResults)/4)

CWMeanIndexValue = round(length(CWResults)/4)

ValidBB_CCW =[];

ValidBB_CW =[];

for i = CCWMeanIndexValue: (CCWMeanIndexValue*3)

    ValidBB_CCW = [ValidBB_CCW; CCWResults(i)];

end

for i = CWMeanIndexValue: (CWMeanIndexValue*3)

    ValidBB_CW = [ValidBB_CW; CWResults(i)];

end
```

```
MeanBallBarCCW = mean(ValidBB_CCW)

MeanBallBarCW = mean(ValidBB_CW)

ValidBallBarData = [ValidBB_CCW;ValidBB_CW];

%need to get radius for MillData?

%if I combine the ideal code part for getting angles from the data pionts

%all I neeed to do then is find the hypotanus

ValidValuesCCW = [];

ValidValuesCW = [];

ValidCorrectedXYZCCW= []; %the

ValidEncodeXYZCCW =[];

ValidCorrectedXYZCW= []; %the

ValidEncodeXYZCW =[];

index = [];
```

```
%need to get valid numbers
for i =12 :length(CorrectedXYZ)
    if CorrectedXYZ(i,2) < ((Radius+0.9)) && i<(length(CorrectedXYZ)/2)
        ValidValuesCCW = [ValidValuesCCW;CorrectedXYZ(i,:)];
        ValidCorrectedXYZCCW = [ValidCorrectedXYZCCW; CorrectedXYZ(i,1)-800,CorrectedXYZ(i,2),CorrectedXYZ(i,3)];
        ValidEncodeXYZCCW = [ValidEncodeXYZCCW; XYZPose(i,1)-800,XYZPose(i,2),XYZPose(i,3)];
        index = [index;i];
    elseif CorrectedXYZ(i,2) < ((Radius+0.9)) && i>(length(CorrectedXYZ)/2)
        ValidValuesCW = [ValidValuesCW;CorrectedXYZ(i,:)];
        ValidCorrectedXYZCW = [ValidCorrectedXYZCW; CorrectedXYZ(i,1)-800,CorrectedXYZ(i,2),CorrectedXYZ(i,3)];
        ValidEncodeXYZCW = [ValidEncodeXYZCW; XYZPose(i,1)-800,XYZPose(i,2),XYZPose(i,3)];
        index = [index;i];
    end
```

```
end


%can not remember what happened here but probs tried to automate eveything
%and did not figure it out till I did it later]


%AllValidValues = [ValidValuesCCW;ValidValuesCW];
% AngCCW = [];
% AngCW = [];
% for i =1 :length(ValidValuesCCW)
%     O_A = ValidValuesCCW(i,1)/ValidValuesCCW(i,2);
%     %disp(ValidValuesCCW(i,1))
%     %commented equations convert to Cobot orintation not Graph
%     if ValidValuesCCW(i,1) < 0 && ValidValuesCCW(i,2) < 0 %both negative
```

```
%         %AngCCW = [AngCCW;360-((atand(O_A)))];
%         AngCCW = [AngCCW;270-((atand(O_A)))];
%     elseif  ValidValuesCCW(i,1) < 0 && ValidValuesCCW(i,2) > 0 %x- y pos
%         %AngCCW = [AngCCW;180-((atand(O_A)))];
%         AngCCW = [AngCCW;90-((atand(O_A)))];
%     elseif  ValidValuesCCW(i,1) > 0 && ValidValuesCCW(i,2) < 0 %x pos y-
%         %AngCCW = [AngCCW;0-((atand(O_A)))];
%         AngCCW = [AngCCW;270-((atand(O_A)))];
%     else %both are posative
%         %AngCCW = [AngCCW;180-((0+atand(O_A)))];
%         AngCCW = [AngCCW;90-((atand(O_A)))];
%     end
% end
```

```
% for i =1 :length(ValidValuesCW)
%     O_A = ValidValuesCW(i,1)/ValidValuesCW(i,2);
%     %disp(ValidValuesCCW(i,1))
%     if ValidValuesCW(i,1) < 0 && ValidValuesCW(i,2) < 0
%         %AngCW = [AngCW;360-((atand(O_A)))];
%         AngCW = [AngCW;270-((atand(O_A)))];
%     elseif  ValidValuesCW(i,1) < 0 && ValidValuesCW(i,2) > 0
%         %AngCW = [AngCW;180-((atand(O_A)))];
%         AngCW = [AngCW;90-((atand(O_A)))];
%     elseif  ValidValuesCW(i,1) > 0 && ValidValuesCW(i,2) < 0
%         %AngCW = [AngCW;0-((atand(O_A)))];
%         AngCW = [AngCW;270-((atand(O_A)))];
%     else
```

```
%         %AngCW = [AngCW;180-((0+atand(O_A)))];

%         AngCW = [AngCW;90-((0+atand(O_A)))];

%     end

% end




%finds the hypot which is the radius using hypot(x,y)

CorrectedRadiusCCW = [];

EncoderRadiusCCW = [];

CorrectedRadiusCW = [];

EncoderRadiusCW = [];

for i =1 :length(ValidCorrectedXYZCCW)

    CorrectedRadiusCCW = [CorrectedRadiusCCW;hypot(ValidCorrectedXYZCCW(i,1),ValidCorrectedXYZCCW(i,2))];
```

```
    EncoderRadiusCCW = [EncoderRadiusCCW;hypot(ValidEncodeXYZCCW(i,1),ValidEncodeXYZCCW(i,2))];

end

for i =1 :length(ValidCorrectedXYZCW)

    CorrectedRadiusCW = [CorrectedRadiusCW;hypot(ValidCorrectedXYZCW(i,1),ValidCorrectedXYZCW(i,2))];

    EncoderRadiusCW = [EncoderRadiusCW;hypot(ValidEncodeXYZCW(i,1),ValidEncodeXYZCW(i,2))];

end

AllCorrectR = [CorrectedRadiusCCW;CorrectedRadiusCW];

AllPoseR = [EncoderRadiusCCW;EncoderRadiusCW];

CorrectedRadiusMeanCCW = mean(CorrectedRadiusCCW)

EncoderRadiusMeanCCW = mean(EncoderRadiusCCW)

CorrectedRadiusMeanCW = mean(CorrectedRadiusCW)

EncoderRadiusMeanCW = mean(EncoderRadiusCW)
```

```
DataForBar =
[MeanBallBarCCW,CorrectedRadiusMeanCCW,EncoderRadiusMeanCCW,MeanBallBarCW,CorrectedRadiusMeanCW,EncoderRadiusMeanC
W];

%bar(DataForBar)
```

16.6 Hybrid Mill Data collection code

```
%% Codes is to be ran on New CNC PC
clc
clear
close all

%%Aerotech set up
arch = computer('arch');
addpath('x64') %location of pre made functions
handle = A3200Connect;%varible to call the hybrid mill


%axis ALLOW ME TO CONTROL ALL AT ONCE
axes = [0,1,2,3,4,5]; % X,Y,Z,W,C,B


%FFt cut off setting
CutOff =0.00001; %change for cut off
NumScan = 100                           %number of scans for accel
XYZMill = []

%thing to test
    %does the mill apply the right corrective action
    %do the numScan need to be so high?
    %fft cut off level
    %is the FFT doing anything?

%% Hybrid mill set up
AllCollectedData = [];
PosFeedBack =[];
```

```
ProPosCommand =[];
PosError = [];
AxisCurrent =[];
AllMFO = [];
SpindelSpeed =[];
AllSpeedTargets = [];


%Aerotech Notes
% %state = A3200ProgramGetTaskState(handle,1)%finds out what the mill is doing
% %State numbers
%     %0 == Unavailable
%     %1 == Inactive
%     %2 == Idle
%     %3 == ProgramReady
%     %4 == ProgramRunning
%     %5 == ProgramFeedheld
%     %6 == ProgramPaused
%     %7 == ProgramComplete
%     %8 == Error
%     %9 == Queue
%
% %A3200ProgramRun(handle,1,'TEST BALL BAR.PGM')  %line to load and run a
% %program
%
% %how to call all of the data needed
% %PositionFeedback
% % XposFeedback = A3200StatusGetItem(handle, 0, A3200StatusItem.PositionFeedback, 0);
% % YposFeedback = A3200StatusGetItem(handle, 1, A3200StatusItem.PositionFeedback, 0);
% % ZposFeedback = A3200StatusGetItem(handle, 2, A3200StatusItem.PositionFeedback, 0);
% %
```

```
% % ProgramPositionFeedback
% % XProgramPositionFeedback = A3200StatusGetItem(handle, 0, A3200StatusItem.ProgramPositionFeedback, 0);
% % YProgramPositionFeedback = A3200StatusGetItem(handle, 1, A3200StatusItem.ProgramPositionFeedback, 0);
% % ZProgramPositionFeedback = A3200StatusGetItem(handle, 2, A3200StatusItem.ProgramPositionFeedback, 0);
% %
% % PositionError
% % XPositionError = A3200StatusGetItem(handle, 0, A3200StatusItem.PositionError, 0);
% % YPositionError = A3200StatusGetItem(handle, 1, A3200StatusItem.PositionError, 0);
% % ZPositionError = A3200StatusGetItem(handle, 2, A3200StatusItem.PositionError, 0);

%% setting up NI and Accel reading
%connecting to board
accel = daq("ni");
    %S = spindel accel
    S0 = addinput(accel, "cDAQ1Mod1", 0, "Accelerometer");%X
    S1 = addinput(accel, "cDAQ1Mod1", 1, "Accelerometer");%Y
    S2 = addinput(accel, "cDAQ1Mod1", 2, "Accelerometer");%Z
    S0.Sensitivity = 1;
    S1.Sensitivity = 1;
    S2.Sensitivity = 1;
    %T = Table accel
    T0 = addinput(accel, "cDAQ1Mod2", 0, "Accelerometer");%X
    T1 = addinput(accel, "cDAQ1Mod2", 1, "Accelerometer");%Y
    T2 = addinput(accel, "cDAQ1Mod2", 2, "Accelerometer");%Z
    T0.Sensitivity = 1;
    T1.Sensitivity = 1;
    T2.Sensitivity = 1;
    %B = Base Accel
    B0 = addinput(accel, "cDAQ1Mod3", 0, "Accelerometer");%X
    B1 = addinput(accel, "cDAQ1Mod3", 1, "Accelerometer");%Y
    B2 = addinput(accel, "cDAQ1Mod3", 2, "Accelerometer");%Z
```

```
    B0.Sensitivity = 1;
    B1.Sensitivity = 1;
    B2.Sensitivity = 1;

t = 0.00009765625; %why a fixed number ???
SR = 10240;                                 %Sample rate = 10240|5120
SF = 1;                                     % customised units scale factor, units/volt
accel.Rate = SR;                            %sets samperling rate
CorrectedXYZ =[];
OffCorrectedXYZ =[];
AllAccelVales = [];
OffsetVoltageDataTable =[];
OffsetVoltageDataBase =[];
OffsetVoltageDataSpin = [];
AllRawData =[];
AllRawDataBase =[];
AllXValues = [];
AllYValues = [];
AllZValues = [];
AllFFTXvalues =[];
AllFFTYvalues =[];
AllFFTZvalues =[];
AllCalAccelTCP = [];
AllCompairedResults = [];

%% ==== running Offset Code  ====
start(accel,"Continuous")
pause(1)
disp('starting offset')
%RawOffsetData = read(accel,seconds(5));     %collects same amount of scans to removed from attachal data
for CollectOffSetData = 0: 1: 5
```

```
    RawOffsetData = read(accel,"NumScans",NumScan);
    CorrectedOffsetDataTable = [RawOffsetData.cDAQ1Mod1_ai0,RawOffsetData.cDAQ1Mod1_ai1,RawOffsetData.cDAQ1Mod1_ai2];
    CorrectedOffsetDataSpin = [RawOffsetData.cDAQ1Mod2_ai0,RawOffsetData.cDAQ1Mod2_ai1,RawOffsetData.cDAQ1Mod2_ai2];
                                                            CorrectedOffsetDataBase                                 =
[RawOffsetData.cDAQ1Mod3_ai0,RawOffsetData.cDAQ1Mod3_ai1,RawOffsetData.cDAQ1Mod3_ai2];
    OffsetVoltageDataTable = [OffsetVoltageDataTable;CorrectedOffsetDataTable];               %stores data in a uasable
and editable formate with time
    OffsetVoltageDataSpin = [OffsetVoltageDataSpin;CorrectedOffsetDataSpin];              %stores data in a uasable and
editable formate with time
    OffsetVoltageDataBase = [OffsetVoltageDataBase;CorrectedOffsetDataBase];              %stores data in a uasable and
editable formate with time
    %pause(1)
end
MeanOffsetTable = mean(OffsetVoltageDataTable)
MeanOffsetSpin = mean(OffsetVoltageDataSpin)
MeanOffsetBase = mean(OffsetVoltageDataBase)%acts a datum for aceel values
disp('offset done');

writePWMVoltage(a, 'D10', NoCompVolt);%Y axis = 5.00
    writePWMVoltage(a, 'D9', NoCompVolt);%Z axis = 5.001
end
AllAxisVeloFeedback = [];
AllFeedRateCommand = [];
AllPredictions = [];
AllAccelVales = [];
%% ==== runs Ball bar test
disp('About to start test')
state = A3200ProgramGetTaskState(handle,1)%finds out what the mill is doing
%A3200ProgramRun(handle,1,'TEST BALL BAR.PGM')
A3200ProgramStart(handle,1);    %runs loaded program
state = A3200ProgramGetTaskState(handle,1)%finds out what the mill is doing
```

```
while state == 4 %'ProgramRunning'
        %XYZ of Hybrid Mill and all other data
         %pos command XYZ
            %A3200StatusGetItem(controllor, axis, data to collect)
            XProgramPositionFeedback = A3200StatusGetItem(handle, 0, A3200StatusItem.ProgramPositionCommand, 0);
            YProgramPositionFeedback = A3200StatusGetItem(handle, 1, A3200StatusItem.ProgramPositionCommand, 0);
            ZProgramPositionFeedback = A3200StatusGetItem(handle, 2, A3200StatusItem.ProgramPositionCommand, 0);
            ProPosCommand = [ProPosCommand;XProgramPositionFeedback,YProgramPositionFeedback,ZProgramPositionFeedback];

            %error XYZ
            XPositionError = A3200StatusGetItem(handle, 0, A3200StatusItem.PositionError, 0);
            YPositionError = A3200StatusGetItem(handle, 1, A3200StatusItem.PositionError, 0);
            ZPositionError = A3200StatusGetItem(handle, 2, A3200StatusItem.PositionError, 0);
            PosError = [PosError;XPositionError,YPositionError,ZPositionError];

            %Current XYZ
            XCuurentAv = A3200StatusGetItem(handle, 0, A3200StatusItem.CurrentFeedbackAverage, 0);
            YCuurentAv = A3200StatusGetItem(handle, 1, A3200StatusItem.CurrentFeedbackAverage, 0);
            ZCuurentAv = A3200StatusGetItem(handle, 2, A3200StatusItem.CurrentFeedbackAverage, 0);
            AxisCurrent = [AxisCurrent;XCuurentAv,YCuurentAv,ZCuurentAv];

            %shouuld be speed command ?
            XVelo = A3200StatusGetItem(handle, 0, A3200StatusItem.VelocityFeedbackAverage, 0);
            YVelo = A3200StatusGetItem(handle, 1, A3200StatusItem.VelocityFeedbackAverage, 0);
            ZVelo = A3200StatusGetItem(handle, 2, A3200StatusItem.VelocityFeedbackAverage, 0);
            AllAxisVeloFeedback = [AllAxisVeloFeedback;XVelo,YVelo,ZVelo];
            %CoordinatedSpeedTargetActual is feedback on velocity???
            FeedRateCommand = A3200StatusGetItem(handle, 1, A3200StatusItem.CoordinatedSpeedTargetActual, 0);
            AllFeedRateCommand = [AllFeedRateCommand;FeedRateCommand];

            %MFO
```

```
            TempMFO = A3200StatusGetItem(handle, 1, A3200StatusItem.MFO, 0);
            AllMFO = [AllMFO;TempMFO];

            %SpindelSpeeds
            TempSpindelSpeed = 3000* A3200StatusGetItem(handle, 0, A3200StatusItem.AnalogInput0, 0);
            if TempSpindelSpeed < 5000
                TempSpindelSpeed = 0;
            end
            SpindelSpeed = [SpindelSpeed; TempSpindelSpeed];

        %XYZ Pos
        XposFeedback = A3200StatusGetItem(handle, 0, A3200StatusItem.ProgramPositionFeedback, 0);
        YposFeedback = A3200StatusGetItem(handle, 1, A3200StatusItem.ProgramPositionFeedback, 0);
        ZposFeedback = A3200StatusGetItem(handle, 2, A3200StatusItem.ProgramPositionFeedback, 0);
        XYZMill = [XYZMill;XposFeedback,YposFeedback,ZposFeedback];
        PosFeedBack = [PosFeedBack;XposFeedback,YposFeedback,ZposFeedback];

        %All Mill Data
                                                                                AllCollectedData                    =
[AllCollectedData;PosFeedBack,ProPosCommand,PosError,AxisCurrent,AllAxisVeloFeedback,FeedRateCommand,TempMFO,SpindelSpee
d];

        %Displacment values from Accel
        ReadRawData = read(accel,"NumScans",NumScan); %need to use read, also need to do more then one scan
                                                                                RawScanData                         =
[seconds(ReadRawData.Time),ReadRawData.cDAQ1Mod2_ai0,ReadRawData.cDAQ1Mod2_ai1,ReadRawData.cDAQ1Mod1_ai2];
        %gets the accel data from table X and Y then Spin Z
                %could be an idea if running a bit slow to remove un used
                %data piont e.g table Z and Spin XY
                                                                                RawScanDataBase                     =
[seconds(ReadRawData.Time),ReadRawData.cDAQ1Mod3_ai0,ReadRawData.cDAQ1Mod3_ai1,ReadRawData.cDAQ1Mod3_ai2];
```

```
        AllRawData =[AllRawData;RawScanData];
        AllRawDataBase =[AllRawDataBase;RawScanDataBase];
        t = (RawScanData(1,1)-RawScanData(end,1))/NumScan;
        %FFT
        Len = NumScan;
        %X FFT
        fhatX = fft((RawScanData(:,2)-MeanOffsetTable(1)),Len); %compute the FFt
        PSDX = fhatX.*conj(fhatX)/Len;%perser septrum (power per freq)
        freq = 1/(t(1)*Len)*(0:Len); %creates the x axis of freq in Hz (used fro poltting)
        LPlotX = 1:floor(Len/2); %only plot the first half of freq (used fro poltting)
        indicesX = PSDX>CutOff; %finds all large freqs with power
        PSDCleanX = PSDX.*indicesX; %zeros all others
        fhatX = indicesX.*fhatX;%zero out small fourier coeffs in Y
        ffiltX = ifft(fhatX); %inverse FFT for filltered time signal
        %plot frec by PSD to see spectrum
        %Y FFT
        fhatY = fft((RawScanData(:,3)-MeanOffsetTable(2)),Len);
        PSDY = fhatY.*conj(fhatY)/Len;
        LPlotY = 1:floor(Len/2);
        indicesY = PSDY>CutOff;
        PSDCleanY = PSDY.*indicesY;
        fhatY = indicesY.*fhatY;
        ffiltY = ifft(fhatY);
        %Z FFT
        fhatZ = fft((RawScanData(:,4)-MeanOffsetSpin(3)),Len);
        PSDZ = fhatZ.*conj(fhatZ)/Len;
        LPlotZ = 1:floor(Len/2);
        indicesZ = PSDZ>CutOff;
        PSDCleanZ = PSDZ.*indicesZ;
        fhatZ = indicesZ.*fhatZ;
        ffiltZ = ifft(fhatZ);
```

```
%X FFT Base
fhatXBase = fft((RawScanDataBase(:,2)-MeanOffsetBase(1)),Len);
PSDXBase = fhatXBase.*conj(fhatXBase)/Len;
%freq = 1/(t(1)*Len)*(0:Len);
LPlotXBase = 1:floor(Len/2);
indicesXBase = PSDXBase>CutOff;
PSDCleanXBase = PSDXBase.*indicesXBase
fhatXBase = indicesXBase.*fhatXBase;
ffiltXBase = ifft(fhatXBase);
%Y FFT Base
fhatYBase = fft((RawScanDataBase(:,3)-MeanOffsetBase(2)),Len);
PSDYBase = fhatYBase.*conj(fhatYBase)/Len;
freq = 1/(t(1)*Len)*(0:Len);
LPlotYBase = 1:floor(Len/2);
indicesYBase = PSDYBase>CutOff;
PSDCleanYBase = PSDYBase.*indicesYBase;
fhatYBase = indicesYBase.*fhatYBase;
ffiltYBase = ifft(fhatYBase);
%Z FFT Base
fhatZBase = fft((RawScanDataBase(:,4)-MeanOffsetBase(3)),Len);
PSDZBase = fhatZBase.*conj(fhatZBase)/Len;
freq = 1/(t(1)*Len)*(0:Len);
LPlotZBase = 1:floor(Len/2);
indicesZBase = PSDZBase>CutOff;
PSDCleanZBase = PSDZBase.*indicesZBase;
fhatZBase = indicesZBase.*fhatZBase;
ffiltZBase = ifft(fhatZBase);

%collect all values
AllXValues = [AllXValues;RawScanData(:,2)];
```

```
        AllFFTXvalues = [AllFFTXvalues;ffiltX];
        AllYValues = [AllYValues;RawScanData(:,3)];
        AllFFTYvalues = [AllFFTYvalues;ffiltY];
        AllZValues = [AllZValues;RawScanData(:,4)];
        AllFFTZvalues = [AllFFTZvalues;ffiltZ];

        % converts from G to mm/S^2 and applies offset
        %FFT and Offset
        Chan1V = ((ffiltX-ffiltXBase) *9806.65); %X
        Chan2V = (((ffiltY-ffiltYBase)) *9806.65); %Y
        Chan3V = (((ffiltZ-ffiltZBase)) *9806.65); %Z
        AccelData = [Chan1V,Chan2V,Chan3V];
        Chan1VBase = (ffiltXBase *9806.65); %X
        Chan2VBase = (ffiltYBase *9806.65); %Y
        Chan3VBase = (ffiltZBase *9806.65); %Z
        Chan1VBaseCombine = (((ffiltX-ffiltXBase)-MeanOffsetBase(1)) *9806.65); %X

        AccelBaseData = [Chan1VBase,Chan2VBase,Chan3VBase];
        CompareChan1V = [Chan1V,Chan1VBase,Chan1VBaseCombine];
        AllCompairedResults = [AllCompairedResults;mean(CompareChan1V)];

        %X
        velocityX = (cumtrapz(t,Chan1V,1));
        DisplacementX= (cumtrapz(t,velocityX,1));
        %Y
        velocityY = (cumtrapz(t,Chan2V,1));
        DisplacementY= (cumtrapz(t,velocityY,1));
        %Z
        velocityZ = (cumtrapz(t,Chan3V,1));
        DisplacementZ= (cumtrapz(t,velocityZ,1));
```

16.7 COBOT Corrective action code

```
%% new combined code for controling the COBOT and adding the displacment readings

% clc

% clear

close all



%notes on what works

    %Setting speed from code

    %running each sections of code

    %update two targets at a time

    %identifying next targets

    %editing appropeate next targets
```

```
%what needs doing

    %way to identify origanl pos for all targets



%I can not find a way still to smooth the movment between the stages

    %for cobto what would be the best way to measure it ??????????



    %% setting up NI and Accel reading

%connecting to board

accel = daq("ni");

%|Accel 1 for Arm/CNC|Accel for Base|Vibration|

C0 = addinput(accel, "cDAQ1Mod1", 0, "Accelerometer");%X

C1 = addinput(accel, "cDAQ1Mod1", 1, "Accelerometer");%Y
```

```
C2 = addinput(accel, "cDAQ1Mod1", 2, "Accelerometer");%Z

C0.Sensitivity = 1;

C1.Sensitivity = 1;

C2.Sensitivity = 1;

B0 = addinput(accel, "cDAQ1Mod2", 0, "Accelerometer");%X

B1 = addinput(accel, "cDAQ1Mod2", 1, "Accelerometer");%Y

B2 = addinput(accel, "cDAQ1Mod2", 2, "Accelerometer");%Z

B0.Sensitivity = 1;

B1.Sensitivity = 1;

B2.Sensitivity = 1;

NumScan = 800                       %number of scans

%disp(accel)

AllRawData = [];
```

```
AllRawDataBase = [];

%% COBOT set up code

RDK = Robolink;

%robot = RDK.Item('UR10e');

SimSpeed = 7;

Radius = 150; %

%SpeedMM_S = 200

CorrectiveActionOn = 1

%Tran X location 144.889

%ResetHome()



% Notes  on speeds

    % 16.6 = 1000 mm/min
```

% 33.3 = 2000 mm/min

% 50 = 3000 mm/min

% 66.6 = 4000 mm/min

% 83.3 = 5000 mm/min

% 100 = 6000 mm/min

%116.6 = 7000 mm/min

%==== Parts ====

robot = RDK.Item('UR10e',RDK.ITEM_TYPE_ROBOT);%when you know other wise use line below

ToolTCP = RDK.Item('COBOT Ball Bar Mount with Tri'); %get the TCP of tool %RobotiQ HandE Gripper

ToolTCP = RDK.Item('RobotiQ Hand-E Gripper'); %get the TCP of tool

TAcell = RDK.Item('Uni Tri Mount');

```
%==== Pose varibles =====

count = 1;

XYZPose = [];%poseition of robot wrist

XYZToolTCP = [];%poseition of tool tcp

XYZTCPtoBase = [];%TCP in ref to base

OFFXYZPose = [];

X_raw = [0]; %raw acell data X

Y_raw = [0]; %raw acell data Y



%===== Target varibles ======

 ML = RDK.Item('Mid Left');

 MR = RDK.Item('Mid Right');

 MB = RDK.Item('Mid Bottom');
```

```
MT = RDK.Item('Mid Top');

MUL = RDK.Item('Mid Upper Left');

MUR = RDK.Item('Mid Upper Right');

MLL = RDK.Item('Mid Lower Left');

MLR = RDK.Item('Mid Lower Right');

TL = RDK.Item('Top Left');

TR = RDK.Item('Top Right');

TB = RDK.Item('Top Bottom');

TT = RDK.Item('Top Top');

TUL = RDK.Item('Top Upper Left');

TUR = RDK.Item('Top Upper Right');

TLL = RDK.Item('Top Lower Left');

TLR = RDK.Item('Top Lower Right');
```

```
LL = RDK.Item('Low Left');

LR = RDK.Item('Low Right');

LB = RDK.Item('Low Bottom');

LT = RDK.Item('Low Top');

LUL = RDK.Item('Low Upper Left');

LUR = RDK.Item('Low Upper Right');

LLL = RDK.Item('Low Lower Left');

LLR = RDK.Item('Low Lower Right');

L2M = RDK.Item('Low Tran');

M2T = RDK.Item('Top Tran');

TargetOrder =
[LUL,LT,LUR,LR,LLR,LB,LLL,LL,L2M,MR,MLR,MB,MLL,ML,MUL,MT,MUR,MR,M2T,TL,TUL,TT,TUR,TR,TLR,TB,TLL,TL];
```

```
%Home pos's

MLH = ML.Pose();

MRH = MR.Pose();

MBH = MB.Pose();

MTH = MT.Pose();

MULH = MUL.Pose();

MURH = MUR.Pose();

MLLH = MLL.Pose();

MLRH = MLR.Pose();

TLH = TL.Pose();

TRH = TR.Pose();

TBH = TB.Pose();

TTH = TT.Pose();
```

```
TULH = TUL.Pose();

TURH = TUR.Pose();

TLLH = TLL.Pose();

TLRH = TLR.Pose();

LLH = LL.Pose();

LRH = LR.Pose();

LBH = LB.Pose();

LTH = LT.Pose();

LULH = LUL.Pose();

LURH = LUR.Pose();

LLLH = LLL.Pose();

LLRH = LLR.Pose();

L2MH = L2M.Pose();
```

```
M2TH = M2T.Pose();

 %TargetOrder                                                                                                                                =
[LUL,LT,LUR,LR,LLR,LB,LLL,LL,L2M,MR,MLR,MB,MLL,ML,MUL,MT,MUR,MR,M2T,TL,TUL,TT,TUR,TR,TLR,TB,TLL,TL];

 TargetOrderHome(:,:,1) =LULH;

 TargetOrderHome(:,:,2) =LTH;

 TargetOrderHome(:,:,3) =LURH;

 TargetOrderHome(:,:,4) =LRH;

 TargetOrderHome(:,:,5) =LLRH;

 TargetOrderHome(:,:,6) =LBH;

 TargetOrderHome(:,:,7) =LLLH;

 TargetOrderHome(:,:,8) =LLH;

 TargetOrderHome(:,:,9) =L2MH;

 TargetOrderHome(:,:,10) =MRH;
```

```
TargetOrderHome(:,:,11) =MLRH;

TargetOrderHome(:,:,12) =MBH;

TargetOrderHome(:,:,13) =MLLH;

TargetOrderHome(:,:,14) =MLH;

TargetOrderHome(:,:,15) =MULH;

TargetOrderHome(:,:,16) =MTH;

TargetOrderHome(:,:,17) =MURH;

TargetOrderHome(:,:,18) =MRH;

TargetOrderHome(:,:,19) =M2TH;

TargetOrderHome(:,:,20) =TLH;

TargetOrderHome(:,:,21) =TULH;

TargetOrderHome(:,:,22) =TTH;

TargetOrderHome(:,:,23) =TURH;
```

```
TargetOrderHome(:,:,24) =TRH;

TargetOrderHome(:,:,25) =TLRH;

TargetOrderHome(:,:,26) =TBH;

TargetOrderHome(:,:,27) =TLLH;

TargetOrderHome(:,:,28) =TLH;




%====== Corrective Matrix for testing ======

% StartPiontT1 =     [1,0,0,0;

%                    0,1,0,200;

%                    0,0,-1,200;

%                    0,0,0,1];

% CorrectiveMat =   [0,0,0,10;
```

```
%                0,0,0,0;

%                0,0,0,0;

%                0,0,0,0];


XCorrect =[0];

YCorrect =[0];

ZCorrect =[0];


%T1.setPose(StartPiontT1)


%List of programs

StartPiont = RDK.Item('StartPiont');

LS1 = RDK.Item('Low Section 1');
```

```
LS2 = RDK.Item('Low Section 2');

LS3 = RDK.Item('Low Section 3');

LS4 = RDK.Item('Low Section 4');

L2M = RDK.Item('Low to Mid');

MS1 = RDK.Item('Mid Section 1');

MS2 = RDK.Item('Mid Section 2');

MS3 = RDK.Item('Mid Section 3');

MS4 = RDK.Item('Mid Section 4');

M2T = RDK.Item('Mid to Top');

TS1 = RDK.Item('Top Section 1');

TS2 = RDK.Item('Top Section 2');

TS3 = RDK.Item('Top Section 3');

TS4 = RDK.Item('Top Section 4');
```

```
AllMoves = RDK.Item('AllMoves');



ListProg =[LS1,LS2,LS3,LS4,L2M,MS1,MS2,MS3,MS4,M2T,TS1,TS2,TS3,TS4,StartPiont];

%ListProg = [AllMoves,StartPiont];

%% Matlab controling arm

%currently the arm does all the movments just need to make it so that only

%the relavent targets update before the next

%SpeedMM_S = 1000

%setSpeed(robot,SpeedMM_S,-1,-1,-1)

      AllCorrections = [0,0,0];



for n = 1 : length(ListProg)-1

    ListProg(n).RunProgram();
```

```
    %setSpeed(robot,SpeedMM_S,-1,-1,-1)

    %b = ListProg(n).Busy(); %this now returns 0????

    while ListProg(n).Busy



     ReadRawData = read(accel,"NumScans",NumScan); %need to use read, also need to do more then one scan

     RawScanData =
[seconds(ReadRawData.Time),ReadRawData.cDAQ1Mod1_ai0,ReadRawData.cDAQ1Mod1_ai1,ReadRawData.cDAQ1Mod1_ai2];

     RawScanDataBase =
[seconds(ReadRawData.Time),ReadRawData.cDAQ1Mod2_ai0,ReadRawData.cDAQ1Mod2_ai1,ReadRawData.cDAQ1Mod2_ai2];

     AllRawData =[AllRawData;RawScanData];

     AllRawDataBase =[AllRawDataBase;RawScanDataBase];

     TempY_raw = [RawScanData(:,3)-RawScanDataBase(:,3)];

     TempX_raw = [RawScanData(:,2)-RawScanDataBase(:,2)];
```

```
pose = robot.Pose();%seems to give wrist TCP not Tool TCP

CurrentXYZ = [ pose(1,4) pose(2,4) pose(3,4)]; %current Pose cartiesa

%CurrentXYZ = [ 800+pose(1,4) pose(2,4) 74.9+pose(3,4)]; %current World pose

XYZPose = [XYZPose;CurrentXYZ];

if CorrectiveActionOn   == 1 ;

   if n < length(ListProg)-1

      T1 = TargetOrder(n*2+1);

      T2 = TargetOrder((n*2+2));

      T1H = TargetOrderHome(:,:,n*2+1);

      T2H = TargetOrderHome(:,:,n*2+2);

      %test correction avlues

         % XCorrect = randi(10,1,1) *0.1;
```

```
    % YCorrect = randi(10,1,1) *0.1;
    % ZCorrect = randi(10,1,1) *0.1;
%real prediction
    [XCorrect,YCorrect,ZCorrect] = GetCorrection(TempY_raw,TempX_raw);
        % FirstTarPos = T1.Pose();
        % SecodTarPos = T2.Pose();
        %modify matrix
        AllCorrections = [AllCorrections; XCorrect,YCorrect,ZCorrect];
        FirstTarPos = T1H + [0,0,0, XCorrect; 0,0,0, YCorrect; 0,0,0, ZCorrect; 0,0,0,0];
        SecodTarPos = T2H + [0,0,0, XCorrect; 0,0,0, YCorrect; 0,0,0, ZCorrect; 0,0,0,0];
        %apply correction predicted
        T1.setPose(FirstTarPos)
        T2.setPose(SecodTarPos)
```

```
            %[FirstTarPos,SecodTarPos] = Update2Targets(T1,T2,XCorrect,YCorrect,ZCorrect);
            pause(0.1)
          end
        end
        TempX_raw =[];
        TempY_raw =[];
      end
end
% for i = 1 : length(TargetOrderHome)
%     TargetOrder(i).setPose(TargetOrderHome(i))
% end


 ML.setPose(MLH)
```

MR.setPose(MRH)

MB.setPose(MBH)

MT.setPose(MTH)

MUL.setPose(MULH)

MUR.setPose(MURH)

MLL.setPose(MLLH)

MLR.setPose(MLRH)

TL.setPose(TLH)

TR.setPose(TRH)

TB.setPose(TBH)

TT.setPose(TTH)

TUL.setPose(TULH)

TUR.setPose(TURH)

```
TLL.setPose(TLLH)

TLR.setPose(TLRH)

LL.setPose(LLH)

LR.setPose(LRH)

LB.setPose(LBH)

LT.setPose(LTH)

LUL.setPose(LULH)

LUR.setPose(LURH)

LLL.setPose(LLLH)

LLR.setPose(LLRH)

L2M.setPose([     1.000000,     0.000000,     0.000000,   144.888785 ;
    0.000000,    -1.000000,    -0.000000,    -0.000215 ;
    0.000000,     0.000000,    -1.000000,   -38.823000 ;
```

```
     0.000000,     0.000000,     0.000000,     1.000000 ])

 M2T.setPose([     1.000000,     0.000000,     0.000000,  -144.889000 ;

     0.000000,    -1.000000,    -0.000000,     0.000000 ;

     0.000000,     0.000000,    -1.000000,    38.823000 ;

     0.000000,     0.000000,     0.000000,     1.000000 ])

%this gets the corrections to be applied Abs ML will go here

ListProg(length(ListProg)).RunProgram();  %brings the code back to the start piont



%% sub functions to make predictions and apply the corrections

% function [CorX,CorY,CorZ] = GetCorrection (X_raw,Y_raw)

% %     CorX = randi([1 50]);

% %     CorY = randi([1 50]);

% %     CorZ = randi([1 50]);
```

```
%     %equations fro fingin the dynamic error

%     peakamp=max(X_raw);        % Max amplitude value of raw accelerometer signal (x-axis)

%     yp = bandpower(Y_raw);     % total power of the raw accelerometer signal (y-axis)

%     rmssig = rms(X_raw);       % root mean square value (energy) of the raw accelerometer signal (x-axis)

%     kurt= kurtosis(X_raw);     % kurtosis of the raw accelerometer signal (x-axi

%     D_E = (0.0158*peakamp)+(21.8*yp)-(0.362*exp(-1420.0*(0.121-rmssig).^2-16.5*(0.00161*kurt-1).^2))-0.0209; %D_E is a radius

%     CorX = mean(D_E)

%     CorY = mean(D_E);

%     CorZ = 0;

% end

% function UpdateTargets (LOut,LFeed,Top,Bot,ROut,RFeed,XCorrect,YCorrect,ZCorrect,Radius)

%     %Setting up targets

%     LOutPos = [1,0,0,0 + XCorrect; 0,-1,0,1.5 + Radius + YCorrect; 0,0,-1, 0 + ZCorrect; 0,0,0,1];
```

```
%     LFeedPos = [1,0,0,0 + XCorrect; 0,-1,0,0 + Radius + YCorrect; 0,0,-1, 0 + ZCorrect; 0,0,0,1];

%     TopPos = [1,0,0,0 + XCorrect + Radius; 0,-1,0,0 + YCorrect; 0,0,-1, 0 + ZCorrect; 0,0,0,1];

%     BotPos = [1,0,0,0 + XCorrect - Radius; 0,-1,0,0 + YCorrect; 0,0,-1, 0 + ZCorrect; 0,0,0,1];

%     ROutPos = [1,0,0,0 + XCorrect; 0,-1,0,-1.5-+ Radius + YCorrect; 0,0,-1, 0 + ZCorrect; 0,0,0,1];

%     RFeedPos = [1,0,0,0 + XCorrect; 0,-1,0,0 - Radius + YCorrect; 0,0,-1, 0 + ZCorrect; 0,0,0,1];

%     %updates the pose

%     LOut.setPose(LOutPos)

%     LFeed.setPose(LFeedPos)

%     Top.setPose(TopPos)

%     Bot.setPose(BotPos)

%     ROut.setPose(ROutPos)

%     RFeed.setPose(RFeedPos)

% end
```

```
%seems to fully work just need it to make the prediction

function [FirstTarPos,SecodTarPos] = Update2Targets (FirstTar,SecondTar,XCorrect,YCorrect,ZCorrect)

    %will need to know the original pos of target

    %Call atrget pos hear

    FirstTarPos = FirstTar.Pose();

    SecodTarPos = SecondTar.Pose();

    %modify matrix

    FirstTarPos = FirstTarPos + [0,0,0, XCorrect; 0,0,0, YCorrect; 0,0,0, ZCorrect; 0,0,0,0];

    SecodTarPos = SecodTarPos + [0,0,0, XCorrect; 0,0,0, YCorrect; 0,0,0, ZCorrect; 0,0,0,0];

    %apply correction predicted

    FirstTar.setPose(FirstTarPos)
```

```
    SecondTar.setPose(SecodTarPos)


end


function [CorX,CorY,CorZ] = GetCorrection (X_raw,Y_raw)
%     CorX = randi([1 50]);
%     CorY = randi([1 50]);
%     CorZ = randi([1 50]);
    %equations fro fingin the dynamic error
    peakamp=max(X_raw);        % Max amplitude value of raw accelerometer signal (x-axis)
    yp = bandpower(Y_raw);     % total power of the raw accelerometer signal (y-axis)
    rmssig = rms(X_raw);       % root mean square value (energy) of the raw accelerometer signal (x-axis)
    kurt= kurtosis(X_raw);     % kurtosis of the raw accelerometer signal (x-axi
```

```
    D_E = (0.0158*peakamp)+(21.8*yp)-(0.362*exp(-1420.0*(0.121-rmssig).^2-16.5*(0.00161*kurt-1).^2))-0.0209; %D_E is a radius

    CorX = mean(D_E)

    CorY = mean(D_E);

    CorZ = 0;

end

function [HomePose] = ResetHome()

    RDK = Robolink;

    HomePose = cell(1,28,4);

    MLH = RDK.Item('Mid Left');

    MRH = RDK.Item('Mid Right');

    MBH = RDK.Item('Mid Bottom');

    MTH = RDK.Item('Mid Top');

    MULH = RDK.Item('Mid Upper Left');
```

```
MURH = RDK.Item('Mid Upper Right');

MLLH = RDK.Item('Mid Lower Left');

MLRH = RDK.Item('Mid Lower Right');

TLH = RDK.Item('Top Left');

TRH = RDK.Item('Top Right');

TBH = RDK.Item('Top Bottom');

TTH = RDK.Item('Top Top');

TULH = RDK.Item('Top Upper Left');

TURH = RDK.Item('Top Upper Right');

TLLH = RDK.Item('Top Lower Left');

TLRH = RDK.Item('Top Lower Right');

LLH = RDK.Item('Low Left');

LRH = RDK.Item('Low Right');
```

```
LBH = RDK.Item('Low Bottom');

LTH = RDK.Item('Low Top');

LULH = RDK.Item('Low Upper Left');

LURH = RDK.Item('Low Upper Right');

LLLH = RDK.Item('Low Lower Left');

LLRH = RDK.Item('Low Lower Right');

L2MH = RDK.Item('Low Tran');

M2TH = RDK.Item('Top Tran');

TargetOrderHome =
[LULH,LTH,LURH,LRH,LLRH,LBH,LLLH,LLH,L2MH,MRH,MLRH,MBH,MLLH,MLH,MULH,MTH,MURH,MRH,M2TH,TLH,TULH,T
TH,TURH,TRH,TLRH,TBH,TLLH,TLH];

for i = 1 : length(TargetOrder)

    %this is giving me issues as it does no like applying double
```

```
        %into cell(no idea why might just have to make it a big
        %matrixes which I know how to split
        HomePose(1,i) = TargetOrderHome(i).Pose()
    end
end

%%

% %creates the program but MoveC does not work and it wont say why
% %    %all MoveC are not working ?????
% % prog = RDK.AddProgram('TestProgram');
% % prog.MoveL(LOut);
% % prog.MoveL(LFeed)
```

% % prog.MoveC(Bot,RFeed)

% % prog.MoveC(Top,LFeed)

% % prog.MoveC(Bot,RFeed)

% % prog.MoveC(Top,LFeed)

% % prog.MoveL(LOut)

% %

% % %prog.pause(3) unable to add purse but I could have two moves one CCW and

% % %one CW

% %

% % prog.MoveL(LFeed)

% % prog.MoveC(Top,RFeed)

% % prog.MoveC(Bot,LFeed)

% % prog.MoveC(Top,RFeed)

% % prog.MoveC(Bot,LFeed)

% % prog.MoveL(LOut)

%

% %Deletes the created targets

% % LOut.Delete();

% % LFeed.Delete();

% % Top.Delete();

% % Bot.Delete();

% % ROut.Delete();

% % RFeed.Delete();

16.8 Hybrid Mill Corrective Action Code

```
arch = computer('arch');

addpath('x64')

handle = A3200Connect;




%Notes ------

%features of code

    % RTD

    %accel collection rate is good

    %corrective action

    %time

    %analog Volt input
```

%MFO

%feedback

%error

%current

%Command

%Spindel speed

%things to address

%adding prediction

%ardiuno update rate

%biggest bottle neak is the update rate for the srfiuno on all

%three axis as it goes from 12hz to 6hz h

```
    %seems I have to chnage the uno for another board to get coms
    %faster


clc
 MachineTime = tic;


 %cheeck what parts of the code to run
 AccelOn = 1 %1 = on 0 = off
 RTDOn = 0
 CorrectiveOn = 0 %1 = on 0 = off
 FeedRate = 50 %Need to CHange this "!"""""""""!!!!!""!"!"!"!"
 NumScan = 150                    %number of scans
```

```
%current limit X Y Z W B C

%using this I can determine axis work as a percent using current reading

    %not sure if these are right

CurrentLimit = [16.761,16.761,16.761,9.727,10];



%need to add part to load or genarate the tool path,

%filepath = 'CDS Final Test V3.nc';

filepath = 'BallBarCorrectiveTest.nc' %gcode toolpath

% fiel for ball bar test = 'BallBarCorrectiveTest.nc'

dist_res = 0.01;

angle_res = 0.01;



% try
```

```
%     CommandToolpath = struct2array(load("ToolPathCDSFinalTestV3.mat","toolPath"));

%     disp('loaded Command Path')

% catch

    disp("Creating Command Path ")

    %CommandToolpath = GCodeReader(filepath,dist_res,angle_res)

    GCodeReaderCW

    CommandToolpath = toolPath;

% end

PosIndex = 1;




%things to get all through a3200datacollection or A3200StatusGetItem
```

```
%     pos XYZ

%     Command XYZ

%     Error XYZ

%     Current XYZ

%     MFO

%     Spindel Speed



%varibles to collect all results

AllCollectedData = [];

PosFeedBack =[];

ProPosCommand =[];

PosError = [];

AxisCurrent =[];
```

```
AllMFO = [];

SpindelSpeed =[];

AllSpeedTargets = [];

AllPredictions = [];

AllYFeat = [];

AllZFeat = [];



%RTD Set-Up

if RTDOn == 1

PionterAmount = 50; %chnage the amount of displayed data

%testing at 3000 -50mm test 50 display smaples = 228 over 31.9sec

%testing at 3000 -50mm test 25 display smaples = 228 over 31.9sec

%testing at 3000 -50mm test 10 display smaples = 228 over 31.9sec
```

```
pionter = 2;

i = 1;

CollectingDataX = zeros(1,PionterAmount);

CollectingDataY = zeros(1,PionterAmount);

CollectingDataZ = zeros(1,PionterAmount);

AX = animatedline('Marker','o','Color','B');

AY = animatedline('Marker','o','Color','G');

AZ = animatedline('Marker','o','Color','R');

%legend('Blue = X','Green = Y','Red = Z','0 - line')

title('Real Time Errors')

xlabel('Data Point')

ylabel('Error (mm)')

X = (1:PionterAmount);
```

```
yline(0,'k--');

legend('Blue = X','Green = Y','Red = Z','0 - line')



addpoints(AX,X,CollectingDataX);

addpoints(AY,X,CollectingDataY);

addpoints(AZ,X,CollectingDataZ);

drawnow

end



if AccelOn == 1

    % accel raw data collection set up for all three accels

    %connecting to board

    accel = daq("ni");
```

%|Accel 1 for Arm/CNC|Accel for Base|Vibration|

%S = spindel accel

S0 = addinput(accel, "cDAQ1Mod1", 0, "Accelerometer");%X

S1 = addinput(accel, "cDAQ1Mod1", 1, "Accelerometer");%Y

S2 = addinput(accel, "cDAQ1Mod1", 2, "Accelerometer");%Z

S0.Sensitivity = 1;

S1.Sensitivity = 1;

S2.Sensitivity = 1;

%T = Table accel

T0 = addinput(accel, "cDAQ1Mod2", 0, "Accelerometer");%X

T1 = addinput(accel, "cDAQ1Mod2", 1, "Accelerometer");%Y

T2 = addinput(accel, "cDAQ1Mod2", 2, "Accelerometer");%Z

T0.Sensitivity = 1;

```
T1.Sensitivity = 1;

T2.Sensitivity = 1;

%B = Base Accel

B0 = addinput(accel, "cDAQ1Mod3", 0, "Accelerometer");%X

B1 = addinput(accel, "cDAQ1Mod3", 1, "Accelerometer");%Y

B2 = addinput(accel, "cDAQ1Mod3", 2, "Accelerometer");%Z

B0.Sensitivity = 1;

B1.Sensitivity = 1;

B2.Sensitivity = 1;



%Num samples is critical to speed of system at 50 I get 59898 accel and

%1198 mill data pionts for a 3000 sy ball bar | at 250 smples I get

%59000 accel and 236 mill data pionts
```

```
%disp(accel)



t = 0.00009765625;

SR = 10240;                         %Sample rate = 10240|5120

SF = 1;                             % customised units scale factor, units/volt

%accel.Rate = SR;                       %sets samperling rate

count = 1/SR;                       %used as timer count

CorrectedXYZ =[];

OffCorrectedXYZ =[];

AllAccelVales = [];

OffsetVoltageDataSpin =[];

OffsetVoltageDataTable =[];

OffSetVoltagedataBase = [];
```

```
AllSpinRawData =[];

AllRawDataTable =[];

AllRawDataBaseAccel = [];



% ==== running Offset Code  ====

start(accel,"Continuous")

disp('starting offset')

pause(0.5)

%RawOffsetData = read(accel,seconds(5));     %collects same amount of scans to removed from attachal data

for CollectOffSetData = 0: 1: 2

    %[TestData,Tstamp,~] = read(accel.ScansAvailableFcnCount,"NumScans",NumScan, "OutputFormat", "Matrix");

    %no idea why this is not working might need to go look at much

    %older versions??
```

```
        RawOffsetData = read(accel,"NumScans",NumScan);
        CorrectedOffsetDataSpin = [RawOffsetData.cDAQ1Mod1_ai0,RawOffsetData.cDAQ1Mod1_ai1,RawOffsetData.cDAQ1Mod1_ai2];
        CorrectedOffsetDataTable = [RawOffsetData.cDAQ1Mod2_ai0,RawOffsetData.cDAQ1Mod2_ai1,RawOffsetData.cDAQ1Mod2_ai2];
        CorrectedOffsetDataBase = [RawOffsetData.cDAQ1Mod3_ai0,RawOffsetData.cDAQ1Mod3_ai1,RawOffsetData.cDAQ1Mod3_ai2];
        OffsetVoltageDataSpin = [OffsetVoltageDataSpin;CorrectedOffsetDataSpin];          %stores data in a uasable and editable formate with
time
        OffsetVoltageDataTable = [OffsetVoltageDataTable;CorrectedOffsetDataTable];       %stores data in a uasable and editable formate with
time
        OffSetVoltagedataBase = [OffSetVoltagedataBase;CorrectedOffsetDataBase];
        pause(0.5)
    end
    MeanOffsetSpin = mean(OffsetVoltageDataSpin)
    MeanOffsetTable = mean(OffsetVoltageDataTable)%acts a datum for aceel values
```

```
    MeanOffSetbase = mean(OffSetVoltagedataBase)

    disp('offset done');

    pause(1);

end




if CorrectiveOn == 1

    %calls drive corrective command

    A3200CommandExecute(handle,1,'DRIVECMD X, 150, 1, extractsingle(25000.0), extractsingle(5.0), 0, 0, 0, 0');

    A3200CommandExecute(handle,1,'DRIVECMD Y, 150, 2, extractsingle(25000.0), extractsingle(5.0), 0, 0, 0, 0');

    A3200CommandExecute(handle,1,'DRIVECMD Z, 150, 3, extractsingle(25000.0), extractsingle(5.0), 0, 0, 0, 0');

    %after axis , 150 , Analog Input , Steps per volt , Offset
```

```
try %try allows it to connect but if already no error
    a = arduino('COM8', 'Uno');
end
NoCompVolt = 2.5;
MaxVolt = 4;
MinVolt = 1;
XCorrect = 0;
YCorrect = 0;
ZCorrect = 0;
XPrediction = 0; %these prediction values = um
YPrediction = 0; %these prediction values = um
ZPrediction = 0; %these prediction values = um
%these will save the voltage inputs in so corrections made can be saved
```

```
    AllXCorrections = [];

    AllYCorrections = [];

    AllZCorrections = [];

    AllPredictions = [];

    %2.5 = no compensation

    writePWMVoltage(a, 'D11', NoCompVolt);%X axis = 5.00

    writePWMVoltage(a, 'D10', NoCompVolt);%Y axis = 5.00

    writePWMVoltage(a, 'D9', NoCompVolt);%Z axis = 5.001

end



%moves to known start piont to get home location

%A3200CommandExecute(handle,1,'G1 X0 Y0 Z5 F500');

state = A3200ProgramGetTaskState(handle,1); %does not ned to = state btu will help in runnig loops
```

```
%I dont know it works like that

% while state == 4 %'ProgramRunning'

%     disp('Moving to start location')

%     pause(1)

%     state = A3200ProgramGetTaskState(handle,1); %does not ned to = state btu will help in runnig loops

% end



%Gets starting poistion

XStartPos = A3200StatusGetItem(handle, 0, A3200StatusItem.PositionFeedback, 0);

YStartPos = A3200StatusGetItem(handle, 1, A3200StatusItem.PositionFeedback, 0);

ZStartPos = A3200StatusGetItem(handle, 2, A3200StatusItem.PositionFeedback, 0);

StartLocation = [XStartPos,YStartPos,ZStartPos]
```

```
disp('Start Recording')

pause(1.5)

%used for the first last pos in corrective action

XLastPos = XStartPos;

YLastPos = YStartPos;

ZLastPos = ZStartPos;



state = A3200ProgramGetTaskState(handle,1); %does not ned to = state btu will help in runnig loops

%A3200ProgramRun(handle,1,'TEST BALL BAR.PGM')

tic     %starts timer for machining

A3200ProgramStart(handle,1);          %runs loaded program

state = A3200ProgramGetTaskState(handle,1); %does not ned to = state btu will help in runnig loops
```

```
%when the program is running will repeat while loop

while state == 4 %'ProgramRunning'


    %A3200StatusGetItem allows for geting a data piont you need to state

    % function structure is A3200StatusGetItem (in handle, in itemIndex, in itemCode, in itemExtra)

    %item codes can be found in A3200StatusItem.m


    %pos feedback XYZ

    XposFeedback = A3200StatusGetItem(handle, 0, A3200StatusItem.ProgramPositionFeedback, 0);

    YposFeedback = A3200StatusGetItem(handle, 1, A3200StatusItem.ProgramPositionFeedback, 0);

    ZposFeedback = A3200StatusGetItem(handle, 2, A3200StatusItem.ProgramPositionFeedback, 0);

    PosFeedBack = [PosFeedBack;XposFeedback,YposFeedback,ZposFeedback];
```

```
%pos command XYZ

XProgramPositionFeedback = A3200StatusGetItem(handle, 0, A3200StatusItem.ProgramPositionCommand, 0);

YProgramPositionFeedback = A3200StatusGetItem(handle, 1, A3200StatusItem.ProgramPositionCommand, 0);

ZProgramPositionFeedback = A3200StatusGetItem(handle, 2, A3200StatusItem.ProgramPositionCommand, 0);

ProPosCommand = [ProPosCommand;XProgramPositionFeedback,YProgramPositionFeedback,ZProgramPositionFeedback];



%error XYZ

XPositionError = A3200StatusGetItem(handle, 0, A3200StatusItem.PositionError, 0);

YPositionError = A3200StatusGetItem(handle, 1, A3200StatusItem.PositionError, 0);

ZPositionError = A3200StatusGetItem(handle, 2, A3200StatusItem.PositionError, 0);

PosError = [PosError;XPositionError,YPositionError,ZPositionError];



%Current XYZ
```

```
XCuurentAv = A3200StatusGetItem(handle, 0, A3200StatusItem.CurrentFeedbackAverage, 0);

YCuurentAv = A3200StatusGetItem(handle, 1, A3200StatusItem.CurrentFeedbackAverage, 0);

ZCuurentAv = A3200StatusGetItem(handle, 2, A3200StatusItem.CurrentFeedbackAverage, 0);

AxisCurrent = [AxisCurrent;XCuurentAv,YCuurentAv,ZCuurentAv];

TempCurrent = sum(abs([XCuurentAv,YCuurentAv,ZCuurentAv]));



%MFO

TempMFO = A3200StatusGetItem(handle, 1, A3200StatusItem.MFO, 0);

AllMFO = [AllMFO;TempMFO];



%SpindelSpeeds

TempSpindelSpeed = 3000* A3200StatusGetItem(handle, 0, A3200StatusItem.AnalogInput0, 0);

if TempSpindelSpeed < 5000
```

```
    TempSpindelSpeed = 0;
end
SpindelSpeed = [SpindelSpeed; TempSpindelSpeed];

%Accelarometers data collection
if AccelOn == 1 %collects the accel data if connected
    ReadRawData = read(accel,"NumScans",NumScan); %need to use read, also need to do more then one scan
    RawScanDataSpin =
[seconds(ReadRawData.Time),ReadRawData.cDAQ1Mod1_ai0,ReadRawData.cDAQ1Mod1_ai1,ReadRawData.cDAQ1Mod1_ai2];
    RawScanDataTable =
[seconds(ReadRawData.Time),ReadRawData.cDAQ1Mod2_ai0,ReadRawData.cDAQ1Mod2_ai1,ReadRawData.cDAQ1Mod2_ai2];
    RawScanDataBase =
[seconds(ReadRawData.Time),ReadRawData.cDAQ1Mod3_ai0,ReadRawData.cDAQ1Mod3_ai1,ReadRawData.cDAQ1Mod3_ai2];
```

```
    AllSpinRawData =[AllSpinRawData;RawScanDataSpin];

    AllRawDataTable =[AllRawDataTable;RawScanDataTable];

    AllRawDataBaseAccel = [AllRawDataBaseAccel;RawScanDataBase];

end



% RTD

if RTDOn == 1

clearpoints(AX)

clearpoints(AY)

clearpoints(AZ)



X = (i:(i+(PionterAmount-1)));
```

```
CollectingDataX(pionter) = XPositionError;

CollectingDataY(pionter) = YPositionError;

CollectingDataZ(pionter) = ZPositionError;



xlim([i,i+PionterAmount]);

addpoints(AX,X,CollectingDataX);

addpoints(AY,X,CollectingDataY);

addpoints(AZ,X,CollectingDataZ);

i = i+1;

drawnow



pionter = pionter+1;

if pionter == (PionterAmount+1)
```

```
        pionter = 1;
    end
    end
%%
    if CorrectiveOn == 1 %cheeks to run corrective action code
    %Corrective action callculation


%     %find out which way the corrective action needs to be applied
%     %dont know if this works but will test later
%     %!!!!!!!!!
%         SearchRange = PosIndex + 99; %will need and if statment to cheeck its does not over shot the end of code
%         % how to call a selected range of rows and collums
%             %Y = CommandToolpath(1:100 , 1:3)
```

```
%           %(range of rows, range of collums)
%
%           %identifys the nearest pos to commmand line to see if to add or
%           %subtracke prediction
%       CurrentPos = [XposFeedback,YposFeedback,ZposFeedback];
%
%       Range=[CommandToolpath(PosIndex:SearchRange, 1:3)];
%       NearestPosToCommand = knnsearch(Range,CurrentPos,'Distance','cityblock');
%       PosIndex = NearestPosToCommand;
%       GCommandPos = [CommandToolpath(NearestPosToCommand,:)];


    %testing if method to get forward or back over above method
```

```
        %idea to add a small percent on did not work I think I am over

        %thinking it and should just move on

if XLastPos > (XposFeedback)

    XF = 0;

    XB = 1;

elseif XLastPos < (XposFeedback)

    XF = 1;

    XB = 0;

else

    XF = 0;

    XB = 0;

end

if YLastPos > (YposFeedback)
```

```
    YF = 0;
    YB = 1;
elseif YLastPos < (YposFeedback)
    YF = 1;
    YB = 0;
else
    YF = 0;
    YB = 0;
end
if ZLastPos > (ZposFeedback)
    ZF = 0;
    ZB = 1;
    disp('For')
```

```
elseif ZLastPos < (ZposFeedback)
    ZF = 1;
    ZB = 0;
    disp('Back')
else
    ZF = 0;
    ZB = 0;
end


%New X Y Z equations
%so issues is on Y and Z back???
XAcclValues = RawScanDataTable(:,2)- RawScanDataBase(:,2) - MeanOffsetTable(1);
YAcclValues = RawScanDataTable(:,3)- RawScanDataBase(:,3) - MeanOffsetTable(2);
```

```
ZAcclValues = RawScanDataSpin(:,4)- RawScanDataBase(:,4) - MeanOffsetSpin(3);

%X F

X1 = mean(abs(XAcclValues));

X21= FeedRate;

X3 = rms(XAcclValues);

X6 = max(XAcclValues)/rms(XAcclValues);

X7 = kurtosis(XAcclValues);

FXPrediction = (3.23*(X21-0.476)*(1680*X3-15.9)*(0.72*X6-3.52)*(-90.2*X1+2.65*X7-7.47)+0.00837)*XF



%X B

X21 = FeedRate;

X5 = max(XAcclValues)/mean(abs(XAcclValues));

XF6 = max(XAcclValues)/rms(XAcclValues);
```

```
X7 = kurtosis(XAcclValues);

BXPrediction = (0.697*(X21-0.11)*(1.72*X5-11.1)*(0.401*XF6-2.75)*(9.06*X7-27.3)-0.00562)*XB



%Y F

Y21 = FeedRate;

Y5 = max(YAcclValues)/mean(abs(YAcclValues));

Y6 = max(YAcclValues)/rms(YAcclValues);

FYPrediction = (1.71*(5.04-1.2*Y6)*(Y21-0.0217+exp(-293*(1-0.191*Y5)^2))-0.00498)*YF



%Y B

Y21 = FeedRate;

Y7 = kurtosis(YAcclValues);

Y8 = skewness(YAcclValues);
```

```
Y9 = meanfreq(YAcclValues, 1651);

BYPrediction = (-26.5*(1-0.00258*Y9)^2*(7.36-1.9*Y7)*(Y21+0.0212)-2.53*(Y21-0.00926)*(7.13*Y8-2.55)+0.0127)*YB



%Z F

Z21 = FeedRate;

Z5 = max(ZAcclValues)/mean(abs(ZAcclValues));

Z7 = kurtosis(ZAcclValues);

Z8 = skewness(ZAcclValues);

FZPrediction = (0.617-1.47*exp(-2*(Z21-0.208*Z5+0.628)^2-6.11*(Z8-0.291+0.572*exp(-195*(0.308*Z7-1)^2))^2))*ZF



%Z B

Z21 = FeedRate;

Z1 = mean(abs(ZAcclValues));
```

```
Z8  = skewness(ZAcclValues);

BZPrediction = -(0.869*(12.7-1740*Z1)*exp(-24.2*(1-0.000506*Z21)^2-2*exp(-46.2*(-Z8-0.301)^2))-0.0255)*ZB

%AllYFeat = [AllYFeat; Y1, Y6,Y8,Y9,Y10];

%AllZFeat = [AllZFeat; Z4,Z5,Z7,Z9];



XPrediction = FXPrediction + BXPrediction;

YPrediction = FYPrediction + BYPrediction;

ZPrediction = FZPrediction + BZPrediction;

AllPredictions = [AllPredictions;XPrediction YPrediction ZPrediction];%,XF,YF,ZF,XLastPos,XposFeedback];

XLastPos = XposFeedback;

YLastPos = YposFeedback;

ZLastPos = ZposFeedback;
```

```
pause(0.5)% need to remove after testing



%dont know if I will need this so will need to test on the allFestures code
    if XPrediction == -Inf || XPrediction == Inf %added as equation 2 can give this for sum reason ??
        writePWMVoltage(a, 'D11', NoCompVolt);%X
        disp('inf? X')
    elseif XPrediction > 0 %corrective action is posative
        XCorrect = XPrediction * 0.0205 +NoCompVolt;
        writePWMVoltage(a, 'D11', XCorrect);%X
    elseif XPrediction <0 %corrective action is negative
        XCorrect = XPrediction * 0.02045 +NoCompVolt;
        writePWMVoltage(a, 'D11', XCorrect);%X
```

```
else%no data sets it to nothing
    writePWMVoltage(a, 'D11', NoCompVolt);%X
end


%Y corrective action
if YPrediction == -Inf || YPrediction == Inf %added as equation 2 can give this for sum reason ??
    writePWMVoltage(a, 'D10', NoCompVolt);%X
    disp('inf? Y')
elseif YPrediction > 0 %corrective action is posative
    YCorrect = YPrediction * 0.0205 +NoCompVolt;
    writePWMVoltage(a, 'D10', YCorrect);%X
elseif YPrediction <0 %corrective action is negative
    YCorrect = YPrediction * 0.02045 +NoCompVolt;
```

```
        writePWMVoltage(a, 'D10', YCorrect);%X
    else%no data sets it to nothing
        writePWMVoltage(a, 'D10', NoCompVolt);%X
    end


%     Z correction is way out and no idea why so will need to speak with AB
%     about that


%    Z corrective action
    if ZPrediction == -Inf || ZPrediction == Inf %added as equation 2 can give this for sum reason ??
        writePWMVoltage(a, 'D9', NoCompVolt);%X
        disp('inf? Z')
    elseif ZPrediction > 0 %corrective action is posative
```

```
    ZCorrect = ZPrediction * 0.0205 +NoCompVolt;
    writePWMVoltage(a, 'D9', ZCorrect);%X
elseif ZPrediction <0 %corrective action is negative
    ZCorrect = ZPrediction * 0.02045 +NoCompVolt;
    writePWMVoltage(a, 'D9', ZCorrect);%X
else%no data sets it to nothing
    writePWMVoltage(a, 'D9', NoCompVolt);%X
end


%gets voltage in values so corrections made can be calulated after the
%fact
AllXCorrections = [AllXCorrections, A3200StatusGetItem(handle, 0, A3200StatusItem.AnalogInput1, 0)] ;
AllYCorrections = [AllYCorrections, A3200StatusGetItem(handle, 0, A3200StatusItem.AnalogInput2, 0)] ;
```

```
    AllZCorrections = [AllZCorrections, A3200StatusGetItem(handle, 0, A3200StatusItem.AnalogInput3, 0)] ;
    end

    state = A3200ProgramGetTaskState(handle,1); %updates state for code running check

end
MachineTime = toc;

AllCollectedData = [PosFeedBack ProPosCommand PosError AxisCurrent AllMFO SpindelSpeed];
try
AllAccelVales = [AllSpinRawData,AllRawDataTable,AllRawDataBaseAccel];
end
%Time = A3200StatusGetItem(handle, 0, A3200StatusItem.TotalMoveTime , 0)
```

```
%time is not working???

    %note TaskExecutionTime

%can also call JerkCommandRaw, PositionOffset, SpeedTarget SpeedTargetActual  CoordinatedSpeedCommand  CoordinatedSpeedTargetActual

CoordinatedSpeedTarget CoordinatedDistanceRemaining DistanceLog ProgramLineNumber

MTime = ['Total Machine Time = ',num2str(MachineTime)];

disp(MTime)

disp('fin')

%mean(AllPredictions)

%%

% hold on

% plot(AllPredictions(:,1))

% plot(AllPredictions(:,2))

% %plot(AllPredictions(:,4),'g')
```

```
% plot(AllPredictions(:,4),'g')

% plot(AllPredictions(:,5),'m')
```

16.9 Hybrid mill automated Probe Code

```
' Variables

DVAR $Index

'varible for position

DVAR $X1Position[3]

DVAR $X2Position[3]

DVAR $Y1Position[3]

DVAR $Y2Position[3]

DVAR $ZPosition[3]



DVAR $Compensation[3]

DVAR $probeerror
```

'Old Prob Error setting

'$probeerror = 1.923/2

'New Probe Error Setting (14/2/23)

$probeerror = 1.987/2

'creating prob pionts

'loed location (prob pionts are 5mm from expected block location)

'center of the magnet (no idea how to find that yet)

'Z safe away

'XY Tran piont 9Piont to transistion between X and Y sides

'X1 probe piont

'X2 probe piont

'Y1 probe piont

'Y2 probe piont

'Z1 probe piont

'loactions of the key pionts (order = [Load location, Center mag, Safe Z away, XYTran, X1,X2,Y1,Y2,Z])

'need center first as everything else depends on it

$global[0] = X0 Y0 Z0

$global[1] = X5 Y5 Z5

$global[2] = X10 Y10 Z10

$global[3] = X-10 Y-10 Z-10

$global[4] = X-5 Y-5 Z-5

$global[5] = X0 Y0 Z0

```
$global[6] = X5 Y5 Z5

$global[7] = X10 Y10 Z10

$global[8] = X10 Y10 Z10



' Step 1: Activate the probe



' Step 2: Check the error and battery status of the probe

IF ($DI[3].X == 0)

        $DO[2].X = 0

        MSGBOX DF_ICON_CRITICAL + DF_MSGBOX_OKONLY, "The probe is in error state! Please check it carefully."

        END PROGRAM

ENDIF

IF ($DI[5].X == 0)
```

```
        $DO[2].X = 0
        MSGBOX DF_ICON_CRITICAL + DF_MSGBOX_OKONLY, "The probe battery is low! Please replace the battery."
        END PROGRAM
ENDIF


' Step 2: Configure the A3200 probe function
' Set probe mode, low level trigger, and position storage
PROBE MODE PROBE_SW_MODE 0 $Position[0]
' Specify Input 4 as the probe trigger input
PROBE INPUT PROBE_INPUT_DRIVE_DIGITAL X 4
' Monitor the X, Y and Z axes
PROBE ON X Y Z
```

'how prob is triggered

'INCREMENTAL

'LINEAR $aptglob[0] F10

'DWELL 0.1

'IF (TASKSTATUS(1, DATAITEM_TaskStatus0, 0x40000) == 0) ' Probe was triggered

' LINEAR $aptglob[1] F100

' MSGBOX DF_MSGBOX_OKONLY + DF_ICON_INFORMATION, "{#F6 #F6 #F6}", "Probe was triggered at\x0DX: ", $Position[0]+$Compensation[0], "\x0DY: ", $Position[1]+$Compensation[1], "\x0DZ: ", $Position[2]

'i think I just call the $Position[0] to get its location ???

'MSGBOX DF_MSGBOX_OKONLY + DF_ICON_INFORMATION, "{#F6 #F6 #F6}", "Probe was triggered at\x0DX: ", "\x0DY: ", "\x0DZ: ", "\x0DTest", (\x0D) moves the text to next line

```
                    'MSGBOX DF_MSGBOX_OKONLY + DF_ICON_INFORMATION, "Probe results \x0DX1:"," X"," Y","
Z","\x0DX2:"," X"," Y"," Z" 'got a error of to many arguments?
                    'MSGBOX DF_MSGBOX_OKONLY + DF_ICON_INFORMATION, "Probe results \x0DX1: X  Y
Z","\x0DX2:"," X"," Y"," Z",
                    'usign msgbox might be limiting how and what I can do could try MSGDISPLAY 1, "message"
                        'both of these work it seems to not lilke the $aptglob[0] for some reason so just use $global[0] i guess
                        'MSGDISPLAY 1,   $global[0]
                        'MSGDISPLAY 1, "Value",  $global[0]
                'ELSE
                '       MSGBOX DF_ICON_CRITICAL + DF_MSGBOX_OKONLY, "The probe was not triggered!."
                'ENDIF
MSGDISPLAY 1,   $global[0]
MSGDISPLAY 1,   $global[1]
```

```
MSGDISPLAY 1,   $global[2]

MSGDISPLAY 1,   $global[3]

MSGDISPLAY 1,   $global[4]

MSGDISPLAY 1,   $global[5]

MSGDISPLAY 1,   $global[6]

MSGDISPLAY 1,   $global[7]

MSGDISPLAY 1,   $global[8]


'idea for order of opps

        'load point

        'moves to center of mag

        'prob Z

        'move Z safe
```

'move to X1

'move in -Z

'prob X1

'move to X1

'move to X2

'prob X2

'move to X2

'move to Y1 (need to be cafull to be clear of block

'prob Y1

'Move Y1

'move Y2

'prob Y2

'move Y2

```
'Move Z safe

'move load location



END PROGRAM
```

16.10 COBOT pick and place corrective action code

```
%% new combined code for controling the COBOT and adding the displacment readings

% clc

% clear

close all



    %% setting up NI and Accel reading

%connecting to board

accel = daq("ni");

%|Accel 1 for Arm/CNC|Accel for Base|Vibration|

C0 = addinput(accel, "cDAQ1Mod1", 0, "Accelerometer");%X

C1 = addinput(accel, "cDAQ1Mod1", 1, "Accelerometer");%Y
```

```
C2 = addinput(accel, "cDAQ1Mod1", 2, "Accelerometer");%Z

C0.Sensitivity = 1;

C1.Sensitivity = 1;

C2.Sensitivity = 1;

B0 = addinput(accel, "cDAQ1Mod2", 0, "Accelerometer");%X

B1 = addinput(accel, "cDAQ1Mod2", 1, "Accelerometer");%Y

B2 = addinput(accel, "cDAQ1Mod2", 2, "Accelerometer");%Z

B0.Sensitivity = 1;

B1.Sensitivity = 1;

B2.Sensitivity = 1;

NumScan = 800                       %number of scans

%disp(accel)

AllRawData = [];
```

```
AllRawDataBase = [];



%% COBOT set up code

RDK = Robolink;

%robot = RDK.Item('UR10e');

SimSpeed = 7;

Radius = 150; %

%SpeedMM_S = 200

CorrectiveActionOn = 1      %1 == On | 0 == Off

TestToRun = 2               %1 == lin| 2== Joint| 3== Circile

%Tran X location 144.889

%ResetHome()
```

```
% Notes  on speeds
    % 16.6 = 1000 mm/min
    % 33.3 = 2000 mm/min
    % 50 = 3000 mm/min
    % 66.6 = 4000 mm/min
    % 83.3 = 5000 mm/min
    % 100 = 6000 mm/min
    %116.6 = 7000 mm/min


%==== Parts ====
robot = RDK.Item('UR10e',RDK.ITEM_TYPE_ROBOT);%when you know other wise use line below
ToolTCP = RDK.Item('COBOT Ball Bar Mount with Tri'); %get the TCP of tool %RobotiQ HandE Gripper
ToolTCP = RDK.Item('RobotiQ Hand-E Gripper'); %get the TCP of tool
```

```
TAcell = RDK.Item('Uni Tri Mount');



if TestToRun == 1
    Program = RDK.Item('PP Lin');
elseif TestToRun == 2
    Program = RDK.Item('PP Joint');
elseif TestToRun == 3
    Program = RDK.Item('PP Cir');
else
    disp("error in test selection")
end
%==== Pose varibles =====
count = 1;
```

```
XYZPose = [];%poseition of robot wrist

XYZToolTCP = [];%poseition of tool tcp

XYZTCPtoBase = [];%TCP in ref to base

OFFXYZPose = [];

X_raw = [0]; %raw acell data X

Y_raw = [0]; %raw acell data Y



%pick and place experament

    %thinking about it the code is much simplaier I just have the last

    %piont move and then reset as no other pionts matter?



 PickT = RDK.Item('Pick');

 MoveAwayT = RDK.Item('Move away');
```

```
StartPlaceT = RDK.Item('StartPlace');

PlaceMidT = RDK.Item('PlaceMidPiont');

PlaceT = RDK.Item('Place');



%Home piont of last two targets

PlaceH = PlaceT.Pose();

PlaceMidH = PlaceMidT.Pose(); %probs will not use this



XCorrect =[0];

YCorrect =[0];

ZCorrect =[0];

AllCorrections = [0,0,0];
```

```
%so all my code has to do it get new accel data, error predict then apply,
%and reset the home pos at the end
Program.RunProgram();
while Program.Busy
    ReadRawData = read(accel,"NumScans",NumScan); %need to use read, also need to do more then one scan
    RawScanData = [seconds(ReadRawData.Time),ReadRawData.cDAQ1Mod1_ai0,ReadRawData.cDAQ1Mod1_ai1,ReadRawData.cDAQ1Mod1_ai2];
    RawScanDataBase = [seconds(ReadRawData.Time),ReadRawData.cDAQ1Mod2_ai0,ReadRawData.cDAQ1Mod2_ai1,ReadRawData.cDAQ1Mod2_ai2];
    AllRawData =[AllRawData;RawScanData];
    AllRawDataBase =[AllRawDataBase;RawScanDataBase];
    TempY_raw = [RawScanData(:,3)-RawScanDataBase(:,3)];
    TempX_raw = [RawScanData(:,2)-RawScanDataBase(:,2)];
```

```
%       pose = robot.Pose();%seems to give wrist TCP not Tool TCP
%       CurrentXYZ = [ pose(1,4) pose(2,4) pose(3,4)]; %current Pose cartiesa
%       %CurrentXYZ = [ 800+pose(1,4) pose(2,4) 74.9+pose(3,4)]; %current World pose
%       XYZPose = [XYZPose;CurrentXYZ];


    if CorrectiveActionOn   == 1 ;
        [XCorrect,YCorrect,ZCorrect] = GetCorrection(TempY_raw,TempX_raw);


        %testing code
            %XCorrect = 10*rand;
            %YCorrect = 10*rand;
            %ZCorrect = 0;
```

```
        AllCorrections = [AllCorrections; XCorrect,YCorrect,ZCorrect];

        PlaceCorrection = PlaceH + [0,0,0, XCorrect; 0,0,0, YCorrect; 0,0,0, ZCorrect; 0,0,0,0];

        PlaceT.setPose(PlaceCorrection)

        %pause(0.1)

    end

    TempX_raw =[];

    TempY_raw =[];

end

%reset place piont back

PlaceT.setPose(PlaceH)



  function [CorX,CorY,CorZ] = GetCorrection (X_raw,Y_raw)

  %     CorX = randi([1 50]);
```

```
%     CorY = randi([1 50]);

%     CorZ = randi([1 50]);

    %equations fro fingin the dynamic error

    peakamp=max(X_raw);        % Max amplitude value of raw accelerometer signal (x-axis)

    yp = bandpower(Y_raw);     % total power of the raw accelerometer signal (y-axis)

    rmssig = rms(X_raw);       % root mean square value (energy) of the raw accelerometer signal (x-axis)

    kurt= kurtosis(X_raw);     % kurtosis of the raw accelerometer signal (x-axi

    D_E = (0.0158*peakamp)+(21.8*yp)-(0.362*exp(-1420.0*(0.121-rmssig).^2-16.5*(0.00161*kurt-1).^2))-0.0209; %D_E is a radius

    CorX = mean(D_E);

    CorY = mean(D_E);

    CorZ = 0;

end
```