Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 139 (2026) 174–178

www.elsevier.com/locate/procedia

13th CIRP Global Web Conference (CIRPe 2025)

LLM-Powered Multi-Agent System for Automated Error Detection in Remanufacturing Inspection

David Wesner[a], Dominik Koch[a*], Victor Mas[c], Sven Matthiesen[c], Florian Stamer[b], Gisela Lanza[a]

[a] *wbk Institute of Production Science, Karlsruhe Institut of Technology (KIT), Kaiserstraße 12, 76131 Karlsruhe, Germany*
[b] *Leuphana University, Universitätsallee 1, 21335 Lüneburg, Germany*
[cb] *IPEK Insittute of Production Engineering, Karlsruhe Institut of Technology (KIT), Kaiserstraße 12, 76131 Karlsruhe, Germany*

* Corresponding author: *E-mail address:* dominik.koch@kit.edu

Abstract

The shift toward a circular economy and concepts such as the circular factory require inspection processes capable of handling high variability in product conditions and reliably assessing both surface quality and functional performance. While most automation approaches focus on visual surface defect detection, often powered by convolutional neural networks (CNNs), functional tests on test benches remain largely dependent on manual evaluation by experts. This paper introduces a novel approach that integrates Large Language Model (LLM) agents into the inspection process to automatically analyze data from functional tests. A multi-agent system (MAS) is employed to simulate operational states and manage data generation and coordination, while the LLM-based agent interprets functional test data to detect single and combined faults. Using an angle grinder as a representative case study, we evaluate the capability of this framework to classify complex defect patterns with a mean error rate below 30 %. Our approach complements traditional visual inspection by focusing on functional aspects, demonstrating the potential of combining MAS-based simulation with LLM reasoning for more intelligent and data-driven inspection strategies.

*Keywords:*Multi-Modal Large Language Model; Defect Detection; Circular Economy; Quality Assurance; Industrial Automation.

1. Introduction

The transition toward a circular economy demands new approaches for extending product life cycles and efficiently reusing components. A promising concept in this context is the circular factory, where remanufacturing and repair processes are systematically integrated into the production flow [1]. A critical step in this environment is the inspection process, which must handle varying product conditions, evaluate functionality, and determine the optimal treatment path for each component.

While many automation approaches in inspection focus on visual methods for surface defect detection [2], these techniques often fail to capture functional issues that affect the overall performance of a product. Convolutional Neural Networks (CNNs), for example, have achieved impressive results in detecting surface defects and anomalies on images, but they are limited to visual cues and cannot assess functional behavior [3]. Research such as Kaiser et al. addresses the geometric reconstruction of parts using Reinforcement Learning (RL) to optimize inspection paths, yet functional testing remains underrepresented in automated inspection strategies [4]. Functional tests, often performed on test benches, generate complex multivariate data that is typically analyzed manually by experts.

This paper proposes an approach where Large Language Model (LLM) agents are used to analyze functional test data integrated into the inspection process. Rather than replacing

2212-8271
10.1016/j.procir.2025.09.022

visual inspection, this method complements existing approaches by focusing on the interpretation of sensor and signal data obtained during functional checks. A multi-agent system (MAS) is employed to simulate operational scenarios and manage data flows, while an LLM-based agent interprets the test data to classify defects.

The main contributions of this work are:

- Integration of LLM-based agents for automated defect classification based on functional test data.
- Evaluation of single and combined fault scenarios, demonstrating the ability of LLM agents to detect complex fault patterns.
- Application to a realistic use case, using an angle grinder as a representative product with varying operational states.

2. Related Work

Research in remanufacturing increasingly focuses on automating inspection processes to mitigate the inefficiencies of manual procedures, such as high costs and long downtimes [5]. AI-based solutions have become central, with deep learning methods such as CNNs proving highly effective in both visual defect detection and sensor data anomaly recognition [3]. Furthermore, non-destructive testing techniques - such as ultrasonic and eddy-current inspection - are increasingly combined with AI to improve the reliability and speed of defect classification [6].

MAS have emerged as flexible control architectures in complex production and remanufacturing environments, where dynamic product conditions require decentralized decision-making. Hybrid MAS architectures combining centralized optimization with local agent autonomy have demonstrated improved throughput and robustness [7, 8]. RL is also being used to optimize inspection processes, demonstrating how RL can learn efficient next-best-view strategies for adaptive inspection of remanufactured products [4].

Despite these advances, the integration of LLMs as knowledge agents within MAS for inspection tasks remains largely unexplored. The approach presented in this work addresses this gap by combining MAS-driven simulation with LLM-based error detection to create a more flexible and intelligent inspection framework.

3. Methodology

This work integrates functional testing as part of the inspection process by using an LLM-based agent to classify defects based on synthetic test data. Instead of relying on real-world experiments or detailed physical simulations, a dataset was created to represent different operational conditions of an angle grinder. This dataset was designed using assumptions about characteristic patterns in rotation-rate signals, reflecting normal operation, single-fault scenarios, and combined faults.

3.1. Data generation

A synthetic dataset was designed based on a representative rotational-speed profile from experiments on a real angle grinder on a test bench [9], shown in Figure 1. Based on this nominal operating profile, four synthetic reference datasets were generated: lubrication faults were modeled by proportionally scaling speed values with factors between 0.2 and 0.95; bearing faults were represented by superimposing vibration patterns of known amplitude; motor faults were simulated as continuous speed drops with varying decay rates; and gear-tooth failures appeared as periodic, sharply falling peaks. For each fault type, twenty reference curves were generated. Test datasets were created analogously, distinguishing between single-fault cases and superpositions of two or three concurrent fault types.

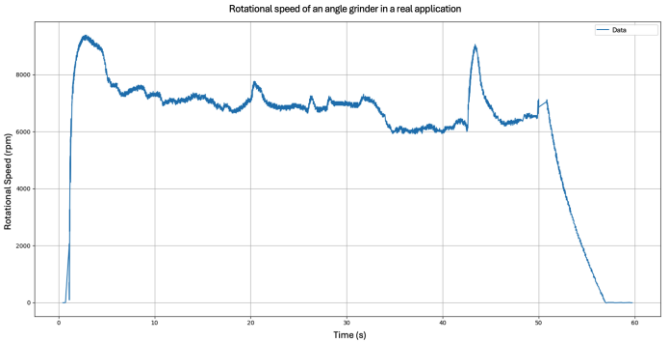

Figure 1: Rotational-speed profile of a real angle grinder on a function testbench [9]

The synthetic dataset allows for controlled experiments, where the fault conditions are known and can be adjusted systematically. While this approach does not capture the full dynamics of a physical system, it provides a consistent and reproducible test environment.

Figure 2 illustrates the overall workflow, from data generation and preprocessing to agent-based analysis and final diagnostic reporting.

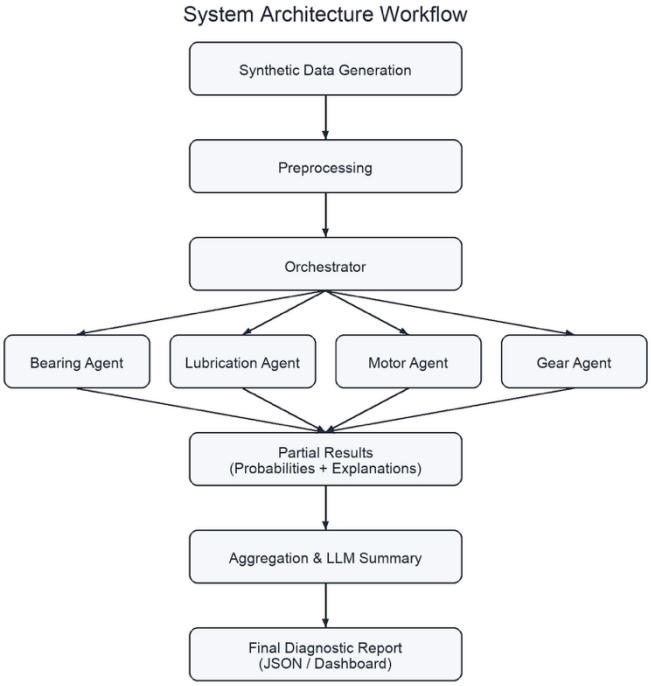

Figure 2: System Architecture Workflow

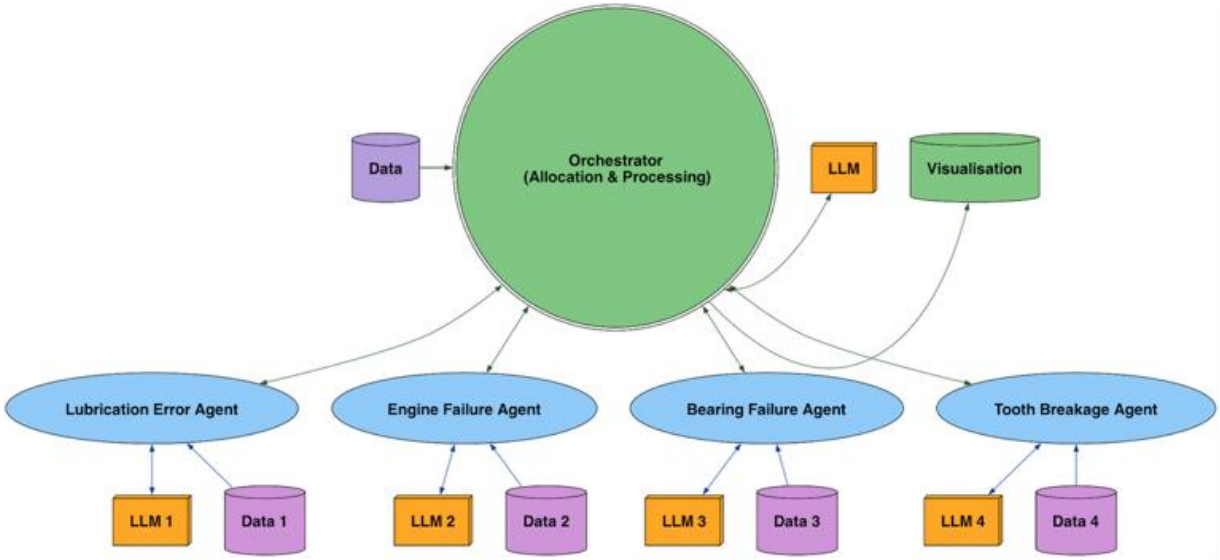

Figure 3: System architecture of the MAS

3.2. Implementation of the Multi-Agent-System

The inspection system was implemented in Python following a modular multi-agent design, enabling structured fault analysis and straightforward extension to new fault types. At the core of the architecture is an Orchestrator class, which initializes four specialized agents – each dedicated to a specific fault category: lubrication defects, bearing damage, gear tooth failures, and motor faults, as shown in Table 1. The developed framework is shown in Figure 3.

Table 1: Damage Agents of the Multi Agent System

Damage Agent	Detection Feature
Bearing damage	Vibration spectra
Lubrication defects	Peak-to-peak amplitudes
Motor faults	Monotonic decay rates
Gear tooth failure	Periodic spike intervals

Upon receiving a JSON file containing time-stamped speed measurements, the Orchestrator uses a *ThreadPoolExecutor* to distribute the input data concurrently to all agents, reducing overall analysis time and utilizing multi-core processing.

Each agent follows an identical pipeline. First, fault-specific reference patterns and feature templates are loaded from precomputed JSON files. For example, the *BearingDamageAgent* loads a list of expected FFT peak frequencies and tolerances, while the *ToothBreakageAgent* references characteristic impact signatures. Next, the incoming sensor data is parsed and transformed into domain-relevant features:

- vibration spectra for bearings,
- peak-to-peak amplitudes for lubrication defects,
- monotonic decay rates for motor faults, or
- periodic spike intervals for gear tooth failures.

These extracted attributes are then integrated into a parameterized prompt template, which instructs the LLM (Sonnet 3.7) [10] to evaluate similarity against the reference patterns, estimate probabilities, and provide a concise technical justification.

Model selection was based on four independent benchmarks - *Thematic Generalization, Multi-Agent Step Race, Creative Story-Writing and Elimination Game* - which were designed to evaluate generalization ability, strategic cooperation, creativity, and reliability. Sonnet 3.7 achieved consistently high scores across all benchmarks, while also offering moderate computational cost and stable response times, making it both economically and technically suitable for the defect classification tasks in this study.

The assembled prompt is submitted to Sonnet 3.7 via the OpenRouter API, with temperature and hyperparameters defined according to the experimental design. The LLM's responses are parsed using regular expressions and JSON wrappers to extract the probability values and the diagnostic text. Each agent then returns a standardized result object containing the fault type, the estimated probability, and a short explanation (e.g., "bearing damage: 92 %, FFT peaks at 400 Hz exceed expected bearing resonance by 6 dB, indicating inner-race damage; recommend bearing replacement").

Once all agents have completed their analyses, the Orchestrator aggregates the individual results. By default, all probabilities are evaluated and considered. Optionally, a configurable threshold can be defined to filter out small results (e.g., below 3%). The results themselves are compiled without applying this filter. If any probability exceeds the chosen threshold, the corresponding fault is flagged. In addition, a secondary prompt can be issued to Sonnet 3.7 to consolidate the findings into a holistic diagnostic summary, leveraging the LLM's reasoning capabilities. The final inspection report includes all probabilities, agent-level explanations, and the LLM's overall assessment, and can be exported as JSON or as a human-readable report for integration into maintenance dashboards.

This multi-agent design offers several advantages:

- Encapsulation: Each fault class is implemented as a separate module, making the system easy to extend.
- Parallelization: Concurrent execution reduces analysis time and scales with available hardware.

- Domain-specific prompts: Tailored templates improve diagnostic accuracy and the clarity of technical explanations.
- Centralized coordination: The Orchestrator ensures both detailed fault-level results and a high-level aggregated view for maintenance engineers.

4. Results

4.1. No fault scenarios

In the absence of errors in the test data, the agents consistently delivered highly accurate results. For each damage scenario, the error estimation remained below 5%, demonstrating a high level of reliability. This level of accuracy was reproducible across repeated evaluations, with only minor deviations of less than 1% observed between individual runs.

4.2. Single-fault scenarios

Without any reference comparison, Sonnet 3.7 proved scarcely able to differentiate fault types: As shown in Table 2, in all four single-fault scenarios, predicted probabilities ranged between 38 % and 72 % and differed only marginally. The mean absolute error (MAE) was at least 48 % and reached 61.6 % for tooth breakage. This homogeneous uncertainty indicates that the model cannot make reliable distinctions without high uncertainty.

Table 2: Error Data for No Reference Data

Scenario	Lubrication	Engine	Bearing	Tooth	MAE
Lubrication	64.30	71.25	48.65	37.90	48.38
Engine	68.41	65.64	49.09	48.41	50.07
Bearing	66.71	72.19	50.71	46.52	58.68
Gear Tooth	64.80	70.30	49.50	38.40	61.55

When appropriate reference data were provided, prediction quality initially improved with higher temperature settings. At temperature = 0.2, MAE ranged from 34.8 % to 50.1 %; at 0.6, it dropped to 24.3 % – 42.4 %, as shown in Table 3.

Table 3: Overview of Error Data at Various Temperatures (temp = 0.2, 0.6, 0.8, 1.0)

Temp	Scenario	Lubrication	Engine	Bearing	Tooth	MAE
0.2	Lubrication	75.00	69.25	25.00	20.00	34.81
	Engine	60.95	65.00	35.71	38.10	42.44
	Bearing	16.25	81.65	12.50	15.00	50.10
	Gear Tooth	20.75	79.25	20.00	25.00	48.75
0.6	Lubrication	75.00	69.25	25.00	20.00	34.81
	Engine	60.95	65.00	35.71	38.10	42.44
	Bearing	16.25	23.50	57.50	15.00	24.31
	Gear Tooth	20.75	79.25	20.00	50.00	42.50
0.8	Lubrication	100.00	84.65	17.50	27.50	32.41
	Engine	86.95	95.81	9.52	26.76	31.86
	Bearing	6.67	14.38	55.19	26.19	23.01
	Gear Tooth	6.85	25.30	12.50	54.20	22.61
1.0	Lubrication	99.75	84.55	36.15	29.65	37.65
	Engine	90.40	94.80	1.00	35.60	33.05
	Bearing	7.76	95.48	16.67	31.43	54.50
	Gear Tooth	7.55	96.00	8.75	23.75	47.14

The best performance occurred at temperature = 0.8, where correct fault types clearly emerged in all four scenarios and MAE fell to 22.6 % – 32.4 %. Only at temperature = 1.0 did differentiation degrade again, as output variability increased and motor faults were overpredicted in almost every case. On average, the error at temperature = 0.8 was 27.5 %.

4.3. Combined-fault scenarios

In the two-fault scenarios at temperature = 0.8, MAE varied widely from 11.3 % (lubrication + motor) to 69.6 % (lubrication + gearbox). Motor faults were systematically overestimated in all combinations, even when absent, reducing discrimination among other classes. The best results were observed in combinations where motor defects inherently generated dominant signals, as shown in Table 4.

Table 4: Combined Error Calculations for Composite Fault Types. Abbreviated scenarios: L = Lubrication, E = Engine Fault, B = Bearing Fault, Z = Gear Tooth Break. Light blue shading marks the fault components according to the target vector.

Fault Type	Scenario	Lubrication	Engine	Bearing	Gear Tooth	MAE
2	L+E	82.77	96.82	7.05	17.59	11.26
2	L+B	15.14	95.68	18.18	15.91	69.57
2	L+Z	73.32	95.45	18.64	25.00	53.94
2	B+E	74.68	95.36	4.55	40.36	53.78
2	Z+E	83.52	95.48	9.52	35.29	40.57
2	B+Z	4.00	95.50	18.18	27.27	63.51
3	B+Z+E	84.05	89.14	43.86	18.23	58.21
3	L+B+Z	58.82	85.09	46.82	15.05	66.10
3	L+Z+E	78.23	83.55	47.05	14.23	42.76
3	L+B+E	81.82	84.32	43.18	20.27	27.74

Three-fault scenarios proved to be challenging: MAE ranged between 27.7 % (lubrication + gearbox + motor) and 66.1 % (lubrication + gearbox + tooth breakage). In these complex cases the model also tended to weight multiple fault types highly, leading to inconsistent overall predictions.

4.4. Prompt optimization and Detection Accuracy

Prompt design provided another lever for performance improvement: a finely balanced, criterion-weighted prompt improved recognition rates by up to 20 % compared to a simple formulation. This finding underscores that both the data foundation and the precise task description to the LLM are critical.

Finally, a case comparison in bearing faults showed that high vibration amplitudes (> 80 % of the reference scale) were reliably detected in over 80 % of cases, whereas subtle patterns (< 40 % amplitude) were detected in only around 50 % of scenarios. This suggests that the model preferentially processes salient signal features.

In summary, the system achieves an error rate of below 30 % only in single-fault scenarios with matching reference data and optimal temperature settings. In the absence of reference data or in multi-fault cases, MAE ranges from 48 % to 70 %. Further optimizations - such as adaptive prompt tuning, integration of additional sensor modalities or multimodal approaches - are required to enhance robustness under real test-stand conditions.

5. Conclusion and Future Work

This paper introduced a defect classification approach that integrates functional testing into the inspection process, utilizing a multi-agent architecture with LLM-based agents. A synthetic dataset of an angle grinder, covering normal, single-fault, and combined-fault scenarios, was used to evaluate the method. The results demonstrate that the LLM-based agents can classify functional test data with a mean error rate below 30 %, while providing transparent and interpretable explanations for each detected fault.

The findings underline the potential of LLMs for analyzing functional test data and supporting inspection processes in circular factory environments. However, the reliance on synthetic data limits the validity of the results, which should therefore be viewed as a proof-of-concept rather than a fully validated industrial solution.

Future work will focus on applying the approach to real test bench data and validating its robustness under realistic noise and variability. Furthermore, combining LLM agents with neural networks trained on time-series data could enhance performance and provide hybrid expert systems, where domain experts can refine and extend knowledge through the interpretability of LLMs and the pattern recognition capabilities of neural networks. This combination has the potential to significantly improve automated functional inspection in industrial settings.

Acknowledgements

Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – SFB 1574 – Project number 471687386.

Declaration of generative AI and AI-assisted technologies in the writing process

During the preparation of this work the author(s) used ChatGPT4o in order to improve language and readability. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the publication.

References

[1] Lanza, G., Deml, B., Matthiesen, S., Martin, M., Brützel, O., & Hörsting, R. (2024). The vision of the circular factory for the perpetual innovative product. *Automatisierungstechnik*, 72(9), 774–788. https://doi.org/10.1515/auto-2024-0012

[2] Koch, D., Kaiser, J.-P., Stamer, F., Stark, R., & Lanza, G. (2025). Enhancing visual inspection in remanufacturing: A reinforcement learning approach with integrated robot simulation. *Procedia CIRP*, 134, 939–944. https://doi.org/10.1016/j.procir.2025.02.228

[3] Valizadeh, M., & Wolff, S. J. (2022). Convolutional Neural Network applications in additive manufacturing: A review. *Advances in Industrial and Manufacturing Engineering*, 4, 100072. https://doi.org/10.1016/j.aime.2022.100072

[4] Kaiser, J.-P., Gäbele, J., Koch, D., Schmid, J., Stamer, F., & Lanza, G. (2024). Adaptive acquisition planning for visual inspection in remanufacturing using reinforcement learning. *Journal of Intelligent Manufacturing*. https://doi.org/10.1007/s10845-024-02478-0

[5] Siddiqi, M. U. R., Ijomah, W. L., Dobie, G. I., Hafeez, M., Pierce, S. G., Ion, W., Mineo, C., & MacLeod, C. N. (2019). Low cost three-dimensional virtual model construction for remanufacturing industry. *Journal of Remanufacturing*, 9, Article 5. https://doi.org/10.1007/s13243-018-0059-5

[6] Kim, Y., Ye, W., Kumar, R., Bail, F., Dvorak, J., Tan, Y., May, M. C., Chang, Q., Athinarayanan, R., Lanza, G., Sutherland, J., Li, X., & Nath, C. (2024). Unlocking the potential of remanufacturing through machine learning and data-driven models—A survey. *Algorithms*, 17(12), 562. https://doi.org/10.3390/a17120562

[7] Dittrich, M.-A., & Fohlmeister, S. (2020). Cooperative multi-agent system for production control using reinforcement learning. *CIRP Annals*, 69(1), 469–472. https://doi.org/10.1016/j.cirp.2020.04.005

[8] Groß, S., Gerke, W., & Plapper, P. (2020). Agent-based, hybrid control architecture for optimized and flexible production scheduling and control in remanufacturing. *Journal of Remanufacturing*, 14, Article 81. https://doi.org/10.1007/s13243-020-00081-z

[9] Gwosch, T. (2019). Antriebsstrangprüfstände zur Ableitung von Konstruktionszielgrößen in der Produktentwicklung handgehaltener Power Tools. *Forschungsberichte IPEK*, 117, 1–158. https://doi.org/10.5445/IR/1000096256

[10] Mazur, L. (2025). LLM Benchmarks. Retrieved April 9, 2025, from https://github.com/lechmazur