APEM journal

Advances in Production Engineering & Management
Volume 20 | Number 2 | June 2025 | pp 224–238
https://doi.org/10.14743/apem2025.2.537

ISSN 1854-6250
Journal home: apem-journal.org
Original scientific paper

Large language models for G-code generation in CNC machining: A comparison of ChatGPT-3.5 and ChatGPT-4o

Šket, K.[a,*], Potočnik, D.[a], Brezocnik, M.[a], Ficko, M.[a], Klančnik, S.[a]

[a]University of Maribor, Faculty of Mechanical Engineering, Maribor, Slovenia

A B S T R A C T

This research explores the viability of producing ISO G-code for 3-axis machining with OpenAI's Chat Generative Pre-Trained Transformer models, particularly ChatGPT-3.5 and the newer GPT-4o. G-code (RS-274-D, ISO 6983) converts human directives into commands that machines can understand, controlling toolpaths, spindle velocities, and feed rates to produce particular aspects of an object. Previously, G-code was generated either by hand or through the use of computer-aided manufacturing (CAM) software along with machine-specific post-processors, both of which may require considerable time and expense. This research aimed to assess the practicality and effectiveness of specific large language models (LLMs) in generating G-code. The assessment took place in three distinct phases on a sample component that required 3-axis machining. These phases included: (1) the self-generated production of G-code for the sample component, (2) the examination of the independently generated G-code in the CAM application, and (3) the recognition and justification of mistakes in the G-code. The outcomes indicated varying abilities with promising findings. This method could accelerate and possibly enhance manufacturing workflows by decreasing reliance on expensive CAM software and specialized knowledge.

A R T I C L E   I N F O

Keywords:
Generative artificial intelligence;
Intelligent manufacturing;
Large language models (LLM);
ChatGPT;
CNC machining;
G-code programming

Corresponding author:
kristijan.sket@um.si
(Šket, K.)

Article history:
Received 9 May 2025
Revised 13 June 2025
Accepted 19 June 2025

1. Introduction

In advanced manufacturing, especially CNC machining, the incorporation of generative AI models like ChatGPT signifies a novel frontier. This integration aligns with the larger trend towards smart manufacturing and machining as outlined in [1, 2]. The rapid advancement of AI technologies, especially data-driven systems, swarm intelligence, and hybrid human-machine systems, symbolizes progress of smart manufacturing [3, 4], in which AI's analytical and predictive power can be applied to enhance and streamline CNC machining processes, advancing the continuous technological revolution in manufacturing characterized by the new age of the Internet and AI [5-7].

The objective of this study was to employ OpenAI's Chat Generative Pre-Trained Transformer, widely referred to as ChatGPT, in CNC machining to explore its ability to identify, comprehend, and produce ISO G-code for milling operations. This study draws inspiration from the extensive adoption of AI in production and manufacturing systems. The research conducted by Hu Li and colleagues [1] offers a technical examination of the elements that are crucial for the acceptance of AI. Bernhard Heiden and colleagues [8] demonstrate how AI can integrate with manufacturing techniques to establish a self-organizing system. Both studies highlight the significance of AI in enhancing material movement and process effectiveness. This emphasis on data analysis aligns with the increasing trend of employing AI to assess and enhance manufacturing parameters.

1.1 Related work

The paper [9] authored by Rane *et al.* reviews ChatGPT, Bard, and various other generative AI technologies and performs a comprehensive examination, featuring a literature review and bibliometric analysis, to identify the prevailing trends and significant factors in the incorporation of AI tools like ChatGPT into manufacturing, pointing out these technologies as a crucial element in the continuous advancement in production engineering.

In a study by Wang *et al*. [10] the authors analyse the use of ChatGPT in the manufacturing industry and evaluate its advantages and disadvantages. It works well with structured and comprehensive answers, yet drawbacks were identified, particularly in the provision of accurate technical expertise and the tendency to generate incorrect information when queries are made from outside the training data. The authors emphasize the importance of human verification of answers for efficient communication.

In their study, Javaid, Haleem and Singh [11] discuss the incorporation of ChatGPT into the framework of Industry 4.0, exploring how ChatGPT can be adapted to automate tasks and examining its applications in Industry 4.0, including improving human-robot collaboration, supporting predictive maintenance, ensuring quality control, and performing big data analytics.

Some extensive research has been conducted into ChatGPT's usage in additive manufacturing (AM) and usage of G-code in that field. Badini *et al*. [12] conducted an evaluation of usability by optimizing the generation of G-codes for fused filament fabrication. ChatGPT's ability to process and optimize suboptimal G-code data has shown its potential to streamline the process. In their article [13], Sriwastwa *et al*. examined the role of ChatGPT in enhancing training for medical 3D printing. Their research indicated that it offers precise and beneficial responses to fundamental inquiries, particularly for novice trainees or newcomers; nonetheless, as the difficulty of the questions rises, particularly for situations demanding practical experience or thorough technical expertise, the shortcomings become increasingly clear. In [14], the authors discuss the use of large language models (LLM) in AM, with a particular focus on their ability to understand and process G-code. The study shows that while models such as GPT-4 and Claude-2 perform excellently in several areas, their ability to comprehensively analyse and capture the complicated geometry of G-code is significantly limited, mainly due to the short length of the context windows.

1.2 Study justification

To authors' knowledge, there are no studies that focus on the use of generative AI models, such as ChatGPT, for the automatic generation, interpretation, and correction of G-code in CNC machining, although there is a variety of research on AI and its applications in manufacturing. This lack of literature highlights the need for a focused study to evaluate the usefulness and drawbacks of these models in actual production scenarios.

This study is the first evaluation of ChatGPT's ability to generate, decode and correct ISO G-code for CNC machining. It explores the potential of ChatGPT to automate key steps in the CNC programming process, in contrast to previous studies that have investigated more general applications of AI in manufacturing. The focus of the study is not on the architectural novelty of the models, but on the empirical evaluation of how these differences affect the performance of G-code programming.

The article is based on experiments that evaluated responses with three main objectives. First, the ability to create G-code with inputs typically used in commercial computer-aided manufacturing (CAM) programs was determined. Second, ChatGPT's understanding of G-code was evaluated by creating a simple program for 3-axis machining. In the third iteration, ChatGPT's ability to recognize errors and attempt to correct the given G-code was evaluated. The goal of this investigation was to assess ChatGPT's capabilities for this type of work and to determine whether it is possible to create G-code for simple CNC machining problems using ChatGPT alone, which could reduce the need for commercial CAM software in the future.

2. Materials and methods

2.1 Used artificial intelligence method

LLMs have transformed natural language processing through transformer-based neural network architectures (TSMs) [15, 16]. The progression of natural language processing encompasses four key stages: statistical language models, neural language models, pre-trained language models, and LLMs, each stage building on the capabilities of the prior generation to improve language understanding and generation [15, 16]. These models have shown potential in areas such as healthcare [15], education [16], and scientific research [17]; yet some challenges remain, including data bias, high computational costs, and ethical considerations.

TSMs are typically divided into two main types: scratch-trained and pre-trained. Models developed from the ground up are tailored for specific tasks, while pre-trained models first undergo extensive training on large text datasets using self-supervised learning methods before they are fine-tuned for specific downstream applications. Instances of pre-trained models are BERT [18] (Bidirectional Encoder Representations from Transformers) and GPT. BERT employs a bidirectional technique to understand the context of words in both directions in a sentence, whereas GPT utilizes a unidirectional approach that focuses on forecasting the next word in a sequence [19, 20].

The functionality and implementation of ChatGPT is complicated and advanced, but the product is a system that can respond to queries and cues like a human [21] and thanks to its scalability, it can manage multiple conversations simultaneously, increasing productivity and reducing the need for human intervention. Its efficiency in processing large amounts of data quickly also saves time.

But ChatGPT also has some significant drawbacks. It can reproduce biases from the training set, potentially promoting discrimination or stereotyping. Additionally, since its knowledge base is limited to training data, it may contain errors for unusual or specialized topics. In this study, the limitations of ChatGPT were tested in the generation of G-code for CNC machining.

2.2 ISO G-code

G-code is a language used to control toolpaths and generate the profiles of an object's features in CNC machining. It is essentially a set of instructions that translate human directives into machine-readable instructions so that CNC machines can operate automatically. The primary aim of G-code is to regulate different aspects of the machining process, including the motion of the cutting tool, feed rates, spindle speeds, and coolant flow, to ensure high precision and efficiency in production, while also enabling safe operation and collision-free machining [22-24].

G-code can be generated automatically with CAM software that uses a virtual 3D model as input to produce the necessary code for machining the part, or it can be manually written. However, creating the G-code manually requires computer and programming skills. Generating the G-code with CAM requires specialized software and a machine-specific post-processor, which can be quite expensive and requires a trained expert to operate. On the other hand, writing the code manually can be time-consuming and repetitive. The use of AI models for the rapid creation of G-code for simple and small series of parts could, therefore, be a cost-effective and fast solution [22].

2.3 Example part

The example part consists of a cube (Fig. 1) measuring 150 × 150 × 150 mm. On its top side, there are four symmetrically arranged holes, each with a diameter of 16 mm, located 25 mm from the edges of the cube and reach a depth of 50 mm. In the centre there is a through-hole with a diameter of 12 mm. This central through-hole is counterbored with a diameter of 25 mm and a depth of 25 mm. In addition, the upper surface must be face-milled by 1 mm.

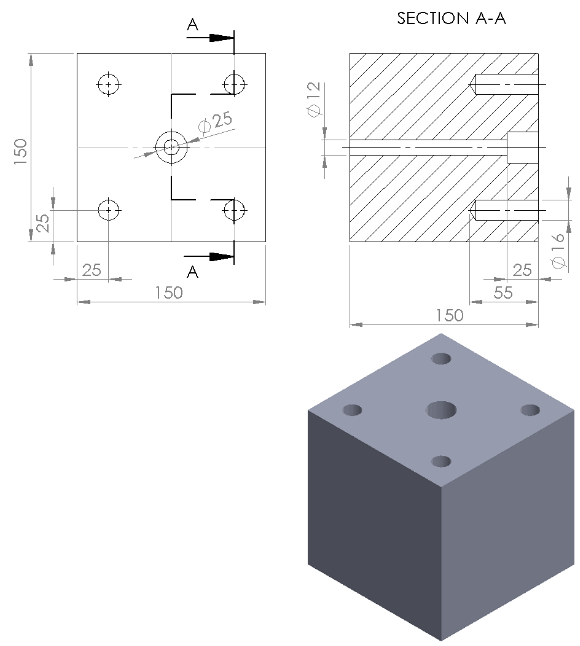

Fig. 1 Design of the example part

2.4 Responses verifying method

To check the answers generated by ChatGPT, the G-code required to produce the example part was created separately in the CAM software Siemens NX version 2312 build 1700 where everything required for a functioning ISO G-code was defined. A part measuring 150 × 150 × 151 mm was defined as the blank and four different tools were provided to create the desired part. Tool characteristics and intended uses are listed in Table 1.

Table 1 Tools properties and intended use

Tool number	Tool name	Feed rate (mm/min)	Surface speed (mm/min)	Operation
T01	Insert cutter ϕ50 mm	250	100	Top face milling
T02	Drill ϕ16 mm	250	100	Drilling holes ϕ16
T03	Drill ϕ12 mm	250	100	Drilling hole ϕ12
T04	Carbide end mill ϕ15 mm	250	100	Pocket and central hole milling

2.5 Queries encoding

The requests were simple and informative, aiming to provide ChatGPT with all relevant information, especially when it was expected to create the G-code for the requested part. The model was encouraged to make suggestions about what information might be missing or could be better presented.

It was decided that the model should first create its own version of the code before the separately created code was fed into the model for debugging and explanation. This way, the model did not have access to any pre-learned data. Additionally, the conversations with the model were performed in the temporary chat, where the input data should not be used for model learning (specified by the service provider).

2.6 G-code preparation

To prepare the G-code readable file for the model, an STL file of the example part and a technical drawing were created. In addition, text descriptions of the part, the tool library, and the capabilities of the CNC machine were added. Features that should be used in the code such as planes, units, feed modes, coordinate systems, and tool lengths compensation modes, etc. were specified to clearly describe what and how the part should be manufactured. The input text is shown in Fig. 2.

I need a complete ISO G-code for 3-axis CNC machining, use this information before you start creating the code:
- Select the XY plane for motion commands,
- set the units to millimeters,
- use absolute positioning mode only,
- activate tool length compensation,
- write the code without comments,
- turn the coolant on and off between operations.

Coordinate system:
- Set the workpiece coordinate system on the top side, lower left corner corresponding to the XY plane.
- Set all coordinates to workpiece coordinate system (always use).

Safety:
- after drilling is completed go to 5 mm above the part and continue from there,
- for milling retract for 1 mm,
- after machining return to reference point.

For the cutting parameters and strategy use:
- Feed rate mode (mm/min),
- 250 millimeters per minute for the feed rate
- 100 surface meters per minute for the automatic calculation of spindle speeds,
- set clockwise rotation for each machining operation,
- add auto calculated spindle speeds,
- use the material data to determine the spindle speed.

Material:
- for the selected part is carbon steel, for the tools: tools are PVD coated.

Geometry of the part description:
- The part is a cube with a size of 150 mm x 150 mm x 150 mm.
- The cube is symmetrical about its central vertical axis.
- The parts top face is milled in its entirety (1st operation).
- There are 4 (four) symmetrically arranged holes on the top face of the cube.
  - o Depth -55 mm.
  - o Each hole has a diameter of 16 mm
  - o Is located 25 mm from the edges of the cube at a depth of 55 mm (2nd operation).
- There is 1 central through hole
  - o diameter of 12 mm (3rd operation).
- There is a central milled blind hole on the top face
  - o diameter of 25 mm.
  - o It has a depth of 25 mm (4th operation).

Description of operations and tools

1st operation:
- Tool no. 1: insert milling cutter, diameter 50 mm (length 34.69 mm flute length 15.4 mm, cutting depth 1 mm).
- Start milling at Z=0 because blank is 1 mm higher and the WCS is at Z=0.
- For milling the top side:
  - o To achieve this, position the tool at the lower left corner (0) of the top face.
  - o Then move the tool along the full length in the X direction.
  - o For each pass in the Y direction move 50 % of the tool diameter (25 mm).
  - o Mill the entire top face of the cube.
  - o Make a final pass at Y=150.

2nd operation:
- Tool no. 2: drill bit, diameter 16 mm (length 143 mm flute length 120 mm), one-handed.
- For drilling four symmetrically arranged holes:
  - o diameter of 16 mm
  - o depth of -55 mm,
  - o located 25 mm from the edges of the cube.
  - o Use a single pass.

3rd operation:
- Tool no. 3 Drill bit diameter 12 mm (length 180 mm flute length 170 mm, single-edged).
- For the central hole:
  - o diameter 12 mm.
  - o The hole needs to be drilled through the entire thickness of the part (-155).

4th operation:
- Tool no. 4 Carbide cutter diameter 15 mm (length 80 mm flute length 30 mm).
- For a 25 mm in diameter blind hole:
  - o The center of the hole should be at X75 Y75.
  - o Note the tool radius (7.5 mm).
  - o Move of center to X80 Y75 before starting circular interpolation.
  - o Preform circular interpolation to mill the entire hole in X and Y direction (diameter 25 mm).
  - o Iterate cutting depth of -5 mm to cut to the hole bottom at -25 mm.

Fig. 2 Text query for G-code generation using ChatGPT-3.5 and GPT-4o

To test the image recognition capabilities of the GPT-4o model and the recognition capabilities of STL files, the text explanation of the designed part in the query was replaced with an STL file of the 3D model and an image of the technical documentation (Fig. 1), as presented in Appendix Fig. A1.

2.7 Debugging G-code

For the requests to debug and repair the created G-code, a part description and the entire G-code with three implemented errors (Table 2) were provided to the model along with the request for ChatGPT to identify and fix them.

For the implemented errors, types were selected that have a major impact on the stability of the process and can lead to damage to the workpiece, tool or machine. The absence of the M3 command can cause the process to start with the spindle switched off, potentially leading to a collision between the tool and the workpiece. The next error is a rapid traverse movement (G0) instead of a feed movement (G1), which means that the tool moves into the workpiece at a much higher speed, which can also result in a collision. In the last error implemented, the spindle speed was set to an impossible speed. In practice, this may cause the process to stop, but it could also result in machining at the maximum speed the machine can deliver, producing an unstable process that can cause serious damage to the workpiece, tool, and machine. The entire query for ChatGPT-3.5 is shown in Fig. A2.

Table 2 Implemented errors

No. of error	Line in code	Error	Correct
1	4	Absence of M3 (spindle on)	M3 after spindle speed set
2	31	G0 (rapid traverse movement)	G1 (movement at feed rate)
3	53	Spindle speed set at 716000 RPM	Spindle speed set at 7160 RPM

2.8 Explaining G-code

For ChatGPT's explanation of the code, the model was provided with a separately created G-code for the specified part and asked to provide a detailed description of each line of code. The request is shown in Fig. 3.

```
%
O0001
G17 G21 G94 G90

(FACE_MILL_ZIGZAG , TOOL : UGT0202_031)

N1 G0 G53 Z762.825
N2 T01 M6
N3 G54
N4 G17 G0 G90 X190. Y20.01 S1061 M3
N5 G43 Z3.4 H1
N6 Z0.2
N7 G94 G1 X165. F250.
N8 X-15.
N9 G2 Y56.67 I0. J18.33
N10 G1 X165.
N11 G3 Y93.33 I0. J18.33
N12 G1 X-15.
N13 G2 Y129.99 I0. J18.33
N14 G1 X165.
N15 X190.
N16 G0 Z3.4
N17 M5
N18 G28 G91 Z0.
N19 G28 X0. Y0.
N20 G90 G28

(DRILLING , TOOL : UGT0301_006)

N21 T02 M6
N22 G54
N23 G17 G0 G90 X25. Y125. S1989 M3
N24 G43 Z11. H2
N25 Z3.
N26 G94 G1 Z-54.807 F250.
N27 G0 Z3.
N28 Z11.
N29 X125.
N30 Z3.
N31 G1 Z-54.807
N32 G0 Z3.
N33 Z11.
N34 Y25.
N35 Z3.
N36 G1 Z-54.807
N37 G0 Z3.
N38 Z11.
N39 X25.
N40 Z3.
N41 G1 Z-54.807
N42 G0 Z3.
N43 Z11.
N44 M5
N45 G28 G91 Z0.
N46 G28 X0. Y0.
N47 G90 G28

(DEEP_HOLE_DRILLING , TOOL : UGT0301_025)

N48 T03 M6
N49 G54
N50 G17 G0 G90 X75. Y75.
N51 G43 Z11. H3
N52 Z3.
N53 S716 M3
N54 G94 G1 Z-4.803 F125. M26
N55 G4 X1.
N56 Z-147. F250.
N57 G4 X1.
N58 Z-154.303 F125.
N59 G4 X1.
N60 M5
N61 M9
N62 G0 Z3.
N63 Z11.
N64 G28 G91 Z0.
N65 G28 X0. Y0.
N66 G90 G28

(HOLE_MILLING , TOOL : UGT0201_133)

N67 T04 M6
N68 G54
N69 G17 G0 G90 X75. Y75. S2122 M3
N70 G43 Z11. H4
N71 Z3.
N72 G94 G3 X80. Y75. I2.5 J0. F250.
N73 X80. Y75. Z-0.75 I-5. J0.
N74 X78.346 Y71.284 Z-4. I-5. J0.
N75 X78.346 Y71.284 I-3.346 J3.716
N76 X75. Y75. I-1.673 J1.858
N77 X80. Y75. I2.5 J0.
N78 X80. Y75. Z-7.75 I-5. J0.
N79 X78.346 Y71.284 Z-11. I-5. J0.
N80 X78.346 Y71.284 I-3.346 J3.716
N81 X75. Y75. I-1.673 J1.858
N82 X80. Y75. I2.5 J0.
N83 X80. Y75. Z-14.75 I-5. J0.
N84 X78.346 Y71.284 Z-18. I-5. J0.
N85 X78.346 Y71.284 I-3.346 J3.716
N86 X75. Y75. I-1.673 J1.858
N87 X80. Y75. I2.5 J0.
N88 X80. Y75. Z-21.75 I-5. J0.
N89 X78.346 Y71.284 Z-25. I-5. J0.
N90 X78.346 Y71.284 I-3.346 J3.716
N91 X75. Y75. I-1.673 J1.858
N92 G0 Z11.
N93 M5
N94 G49
N95 G53 Z762.825
N96 M2
```

Fig. 3 Query for G-code explanation for both models

3. Results

As mentioned earlier, ChatGPT's capabilities with respect to ISO G-code for 3-axis CNC machining were evaluated by verifying its ability to automatically generate the code with the given instructions, detect and correct errors in written code and explain the meaning of the code in detail. In this way, a relevant and practical comparison was achieved by creating realistic scenarios that a CNC programmer could encounter. The tested models GPT-3.5and GPT-4o showed a large discrepancy in understanding both the code and the instructions.

Both models were tested with identical prompts and analysed using Siemens NX CAM software (version 2312 build 1700) for simulation validation. The target geometry, tool set, and all machine constraints were standardized to ensure fairness. The following subsections present the observed differences in model performance.

3.1 G-code generation

When working with the model GPT-3.5, no fully functional code was generated. The closest result was obtained with a semi-functional code, as shown in Fig. 4A. The code contained several errors in both milling operations, such as:

- a collision between the tool and the workpiece,
- movements over the same position,
- circular interpolation in the wrong direction,
- missing sections of the milling operations.

The code was tested in the simulation software Siemens NX version 2312 build 1700. The desired part is shown in Fig. 4B and the resulting part in Fig. 4C. The errors in the code resulted in an unfinished face milling operation, with the turquoise colour representing areas where no face milling operation was performed and the darker blue colour representing areas with a completed face milling operation.

The code also caused a collision and an off-center milled blind hole with an unfinished bottom (see Fig. 4C). In the authors' limited experience, GPT-3.5 is not very useful when working with ISO G-code for 3-axis CNC machining, especially for milling operations. However, it can produce functional code for pure drilling operations.

GPT-4o produced fully functional code (Fig. 5A) that can be used directly for manufacturing on an ISO G-code compatible CNC machine. The code is well organized and follows the given instructions. In addition, a request was sent to the model to display the features generated by the code in 2D (Fig. 5B). The generated code was then tested again in the dedicated CAM software by Siemens NX to verify both the code and the resulting part. The part generated with the GPT-4o code matches the expected design and is shown in Fig. 5C.

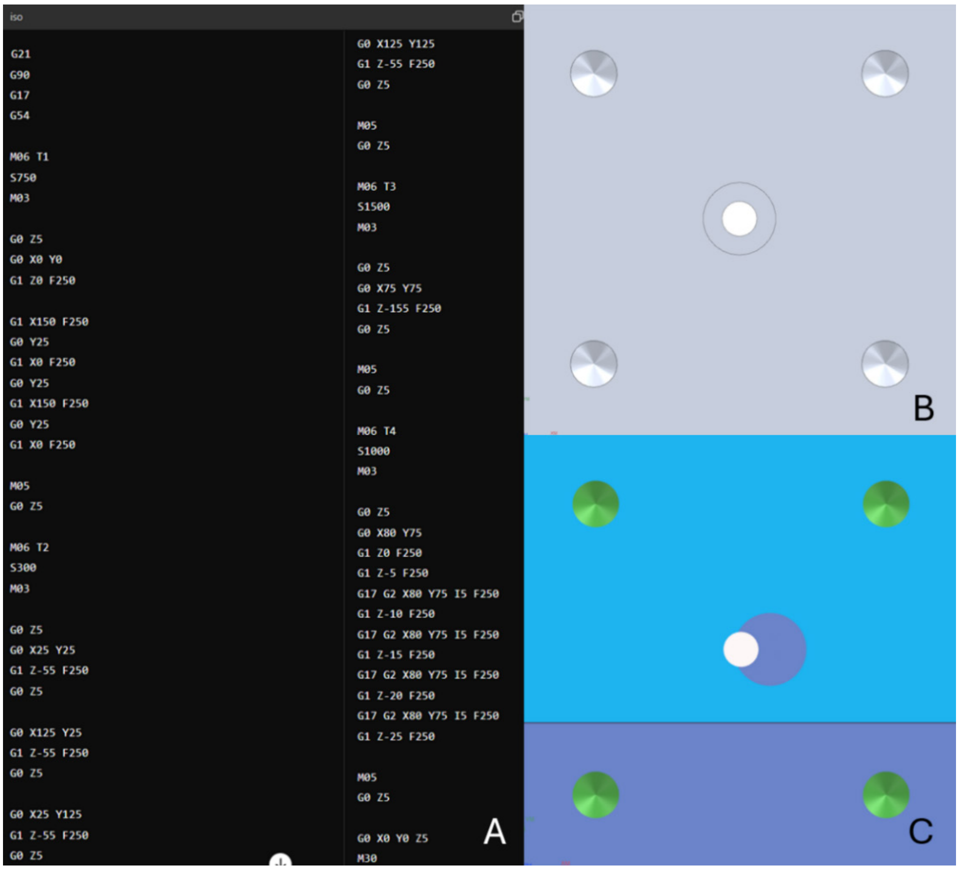

Fig. 4 ChatGPT model 3.5 outputted G-code (A), the desired result of the code (B), and the actual result of the simulated code in Siemens NX version 2312 build 1700 (C)

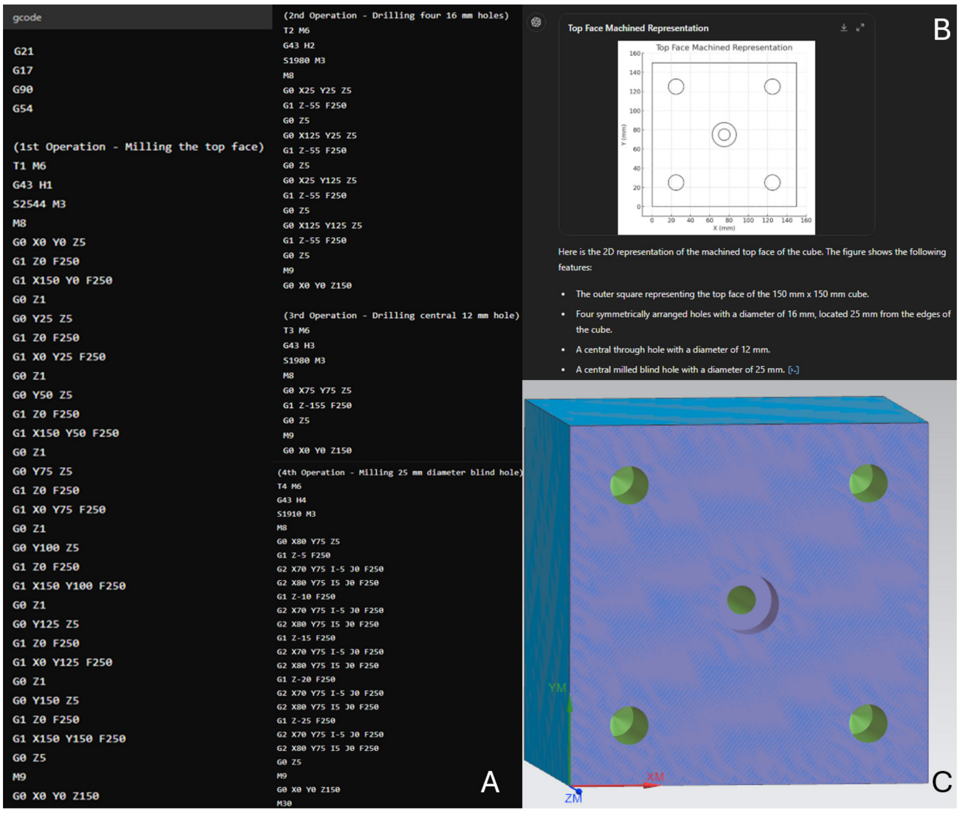

Fig. 5 ChatGPT GPT-4o model output G-code (A), the output depiction (B), and the result of the simulated code in Siemens NX version 2312 build 1700 (C)

3.2 Debugging G-code

A deliberately flawed G-code was presented to both models. The bugs built into the code proved too difficult for model 3.5 to detect. As can be seen on Fig. A3, not only were there no errors detected, but it was even claimed that none were present, and the code should work as expected. This is of course unacceptable and could lead to damage and a failed process. Based on the results the usage of model 3.5 for error detection is not suggested.

In contrast, the ChatGPT-4o model not only successfully identified the errors and implemented the correct syntax, but also explained what the errors were and why they could lead to problems. It additionally warned about other parts of the code that could be redundant or problematic and corrected them.

By focusing only on the errors entered, the model has shown that it understands the code and the process that the code specifies, considering the part, the tool and the machine. It has correctly recognized the missing command to switch on the spindle which causes the machine to start, the G0 command which would result in a collision has been correctly changed to G1 and the set spindle speed has been overridden with a more reasonable value so that the occurrence of errors or damage is mitigated.

The model's responses are shown in Fig. 6. The code was tested again using the Siemens NX simulation tool to check that it still worked as intended, which it did.

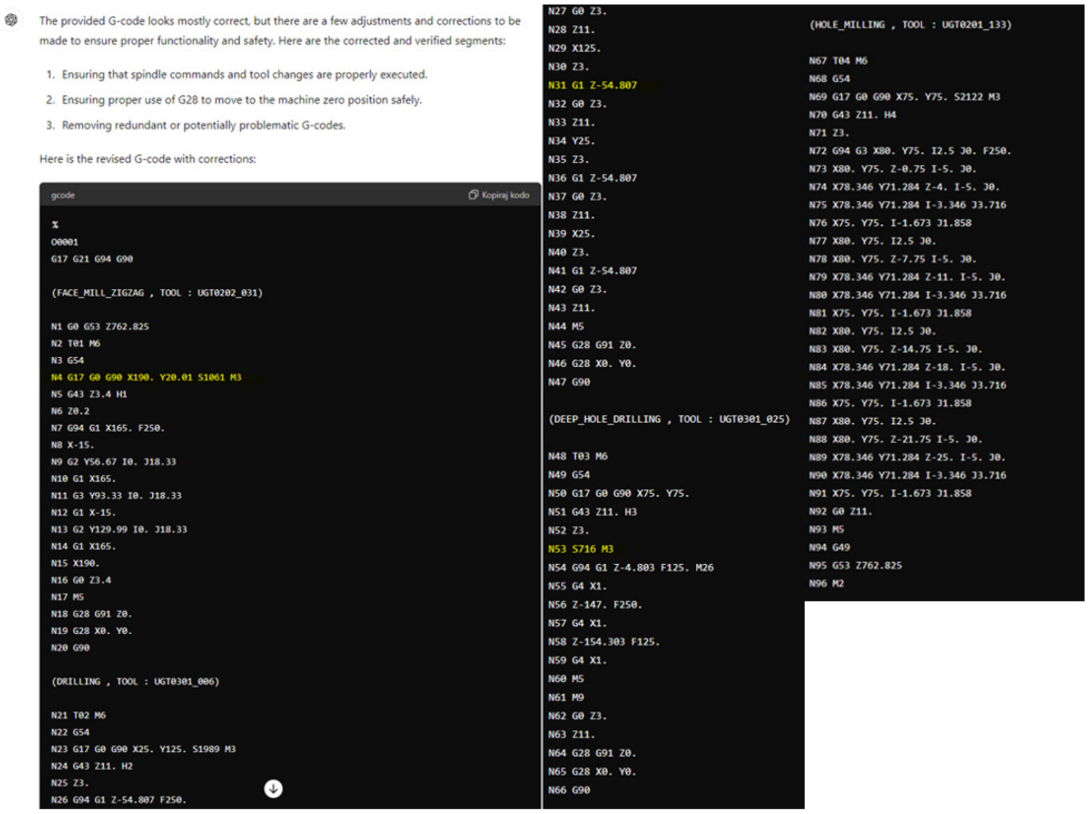

Fig. 6 ChatGPT-4o model error detection capabilities (corrections highlighted in yellow)

3.3 Explaining G-code

When it comes to explaining the meaning of the commands in the code, GPT-3.5 provides only a very limited explanation. Although it was instructed to explain the code line by line, many commands were skipped, and those that were explained are presented very briefly, though correctly. A potential issue is that the model skipped commands related to movements during machining operations, which are important to know. The explanation of the code by GPT-3.5 is shown in Fig. A4.

As in the previous categories, GPT-4o once again outperformed GPT-3.5, this time in code explanation. The model provided a line-by-line explanation of each command in a readable and coherent manner. It correctly recognized different types of commands, such as program start and end, selected planes, units, rapid and work movements, tool selection, coordinate system selection, tool in moves and retraction, machine settings, etc. This demonstrates that the model understands various aspects of CNC machining and can recognize and explain them correctly. Such detailed explanations could be useful for individuals learning ISO G-code or for teaching custom machine learning models. The detailed explanation of the code is shown in Fig. A5.

4. Discussion

The results show that the performance differences between GPT-3.5 and GPT-4o are not only quantitative (i.e., in terms of the number of parameters) but also functional, particularly in the context of ISO G-code. In contrast to previous general AI comparisons, this study introduces a task-specific benchmarking framework that is validated with professional CAM tools. The study provides a basis for understanding LLMs role in CNC programming; however, several critical aspects, including the handling of complex geometries, integration with CAM systems, and specific limitations require further discussed.

4.1 Practical implementation challenges

The application of GPT-4o in CNC machining practise poses various practical difficulties. The conclusion of the study emphasizes the need for professional monitoring and highlights that the models cannot yet completely replace the CAM software. Even small errors in the G-code can lead to costly machine damage, production delays or safety risks and therefore require strict validation processes. In practise, ensuring error-free G-code requires experienced operators to check outputs, which can reduce the appeal of models in demanding production environments.

The reliance on carefully crafted prompts emphasises the importance of input quality. Incomplete or inconsistent prompts can lead to erroneous G-code, especially for users with little CNC knowledge who may struggle to define machining parameters such as feed rates, toolpaths or coordinate systems. This presents a challenge because operators may have varying levels of technical knowledge.

The lack of direct integration with CNC machines or CAM systems complicates implementation. Unlike CAM software, which can seamlessly interface with CAD models and machine controllers, GPT operates as a standalone tool. Users must manually input data and transfer the output to the machining systems, a process that can introduce errors. Integration as a CAM plug-in is possible but would require significant development effort to achieve real-time data transfer and compatibility with different machine controllers.

Additionally, the research assumes that the ISO G-code is compatible with all CNC controls, as most modern controls (e.g. from Fanuc, Siemens or Heidenhain) comply with this standard. However, controls often require machine-specific post-processing to accommodate slight differences in syntax, proprietary M-codes or specified cycles. GPT-4o has effectively generated G-code that has been validated in Siemens NX, but its ability to adapt to control-specific details without direct guidance has yet to be evaluated. Some controls may require additional commands to change tools or specific formatting for coordinate systems.

4.2 Model performance for complex geometries

ChatGPT-3.5 generated a partially functional G-code, whereas GPT-4o generated a functional G-code, demonstrating its improved ability to understand and generate machining instructions. Nevertheless, both models struggle with complex geometries, such as free-form surfaces or complex toolpaths, which were not investigated in this study.

Complicated shapes require accurate toolpath calculations and an understanding of complex machining dynamics. The research focuses on a basic 3-axis component, raising concerns about the models' ability to scale to multi-axis machining (e.g., 4- or 5-axis), where tool orientation and simultaneous multi-axis movements add complexity. Both models, particularly ChatGPT-3.5, are likely to struggle in such scenarios due to their reliance on text-based input and limited capacity to process complex spatial information.

4.3 Specific LLM limitations

Specific LLM constraints, such as the size of the context window have a significant impact on the performance of the G-code. The context window, which determines the amount of text an LLM can process at once, poses a challenge for large G-code files or complex parts with thousands of lines of code.

The scope of the training data also affects performance. While GPT-4o's training likely includes diverse text data, it may lack comprehensive CNC machining datasets, especially for specialised processes or proprietary control syntax. This gap can lead to errors in generating G-code for niche applications or in interpreting ambiguous prompts. For example, the study shows that ChatGPT-3.5 had difficulty with milling operations, likely due to insufficient training in CNC-specific terminology and processes. The improved performance of GPT-4o suggests a broader training dataset, yet its limitations become apparent with advanced toolpath strategies that require fine-tuned prompts to specify parameters accurately. Addressing these limitations may require fine-tuning the LLMs on CNC-specific data sets or enlarging the context windows to handle larger G-code files.

4.4 Future directions

Future research should focus on several key factors to improve the application of LLMs in CNC machining. Evaluating GPT-4o with complex geometries and multi-axis machining scenarios would reveal its scalability and limitations. Secondly, establishing standardised metrics such as error rates, generation time and simulation success would allow a thorough comparison with CAM software. Thirdly, implementing CAM plugins, could connect customised LLM applications with the industry. Finally, LLM-specific constraints, such as the size of the context window and the amount of training data, may improve performance in advanced machining activities through fine-tuning or customised data sets.

In summary, GPT-4o has potential for creating, debugging, and clarifying G-code for basic 3-axis CNC machining and serves as a cost-effective alternative to conventional CAM software. Nevertheless, the problems associated with complicated shapes, challenging toolpath techniques and LLM limitations require further research and development. Under expert supervision and potential integration into CAM systems, GPT-4o could become an indispensable tool in manufacturing, particularly for small-batch production or for educational purposes.

5. Conclusion

This study has demonstrated the functionality of ChatGPT-3.5 and ChatGPT-4o in CNC machining using ISO G-code. Notable differences in performance were observed when evaluating when evaluating the ability of the AI models to generate, interpret and correct ISO G-code. ChatGPT-3.5 showed limitations, particularly in identifying errors and explaining the code, frequently skipping lines and providing terse and uninformative descriptions. It also struggled with milling, although it showed some ability with simpler tasks such as drilling.

In comparison, ChatGPT-4o produced fully functional ISO G-code for the example part, demonstrating its capability for applications with simple geometries. It showed an improved understanding of the code, successfully detecting and correcting errors while providing clear and thorough explanations for each line of code. This makes it a useful resource for learning and teaching G-code programming, as well as an additional verification method when code simulation is not possible. Despite its advances, ChatGPT-4o cannot yet replace traditional CAM programming, especially for complex operations. Its limitations, such as the requirement for text-only input, can lengthen the information input process. However, for simple operations and in situations where CAM software is not available, it can significantly reduce the time required to manually write the G-code.

In conclusion, ChatGPT, especially the GPT-4o model, has some potential to improve G-code programming, however, it still requires expert supervision. Subsequent studies should aim to improve the functionalities for more complicated machining operations and increase the integration of these AI models into current CAM systems.

Funding and acknowledgment

The authors acknowledge the financial support from the Slovenian Research and Innovation Agency (research core funding No. P2-0157).

Declaration of competing interests

The authors declare that they have no known competing financial interests or personal relationships that could influence the work in this article.

References

[1] Li, B.-H., Hou, B.-C., Yu, W.-T., Lu, X.-B., Yang, C.-W. (2017). Applications of artificial intelligence in intelligent manufacturing: A review, *Frontiers of Information Technology & Electronic Engineering*, Vol. 18, 86-96, doi: 10.1631/FITEE.1601885.
[2] Yang, T., Yi, X., Lu, S., Johansson, K.H., Chai, T. (2021). Intelligent manufacturing for the process industry driven by industrial artificial intelligence, *Engineering*, Vol. 7, No. 9, 1224-1230, doi: 10.1016/j.eng.2021.04.023.
[3] Tao, F., Qi, Q., Liu, A., Kusiak, A. (2018). Data-driven smart manufacturing, *Journal of Manufacturing Systems,* Vol. 48, Part C, 157-169, doi: 10.1016/j.jmsy.2018.01.006.
[4] Davis, J., Edgar, T., Graybill, R., Korambath, P., Schott, B., Swink, D., Wang, J., Wetzel, J. (2015). Smart manufacturing, *Annual Review of Chemical and Biomolecular Engineering,* Vol. 6, 141-160, doi: 10.1146/annurev-chembioeng-061114-123255.
[5] Wan, J., Li, X., Dai, H.-N., Kusiak, A., Martinez-Garcia, M., Li, D. (2021). Artificial-intelligence-driven customized manufacturing factory: Key technologies, applications, and challenges, *Proceedings of the IEEE*, Vol. 109, No. 4, 377-398, doi: 10.1109/JPROC.2020.3034808.
[6] Lee, J., Davari, H., Singh, J., Pandhare, V. (2018). Industrial artificial intelligence for industry 4.0-based manufacturing systems, *Manufacturing Letters,* Vol. 18, 20-23, doi: 10.1016/j.mfglet.2018.09.002.
[7] Yao, X., Zhou, J., Zhang, J., Boer, C.R. (2017). From intelligent manufacturing to smart manufacturing for industry 4.0 driven by next generation artificial intelligence and further on, In: *Proceedings of 2017 5th International Conference on Enterprise Systems (ES),* Beijing, China, 311-318, doi: 10.1109/ES.2017.58.
[8] Heiden, B., Alieksieiev, V., Volk, M., Tonino-Heiden, B. (2021). Framing artificial intelligence (AI) additive manufacturing (AM), *Procedia Computer Science,* Vol. 186, 387-394, doi: 10.1016/j.procs.2021.04.161.
[9] Rane, N., Choudhary, S., Rane, J. (2024). Intelligent manufacturing through generative artificial intelligence, such as ChatGPT or Bard, *SSRN Electronic Journal*, doi: 10.2139/ssrn.4681747.
[10] Wang, X., Anwer, N., Dai, Y., Liu, A. (2023). ChatGPT for design, manufacturing, and education, *Procedia CIRP*, Vol. 119, 7-14, doi: 10.1016/j.procir.2023.04.001.
[11] Javaid, M., Haleem, A., Singh, R.P. (2023). A study on ChatGPT for industry 4.0: Background, potentials, challenges, and eventualities, *Journal of Economy and Technology*, Vol. 1, 127-143, doi: 10.1016/j.ject.2023.08.001.
[12] Badini, S., Regondi, S., Frontoni, E., Pugliese, R. (2023). Assessing the capabilities of ChatGPT to improve additive manufacturing troubleshooting, *Advanced Industrial and Engineering Polymer Research*, Vol. 6, No. 3, 278-287, doi: 10.1016/j.aiepr.2023.03.003.
[13] Sriwastwa, A., Ravi, P., Emmert, A., Chokshi, S., Kondor, S., Dhal, K., Patel, P., Chepelev, L.L., Rybicki, F.J., Gupta, R. (2023). Generative AI for medical 3D printing: A comparison of ChatGPT outputs to reference standard education, *3D Printing in Medicine,* Vol. 9, Article No. 21, doi: 10.1186/s41205-023-00186-8.
[14] Jignasu, A., Marshall, K., Ganapathysubramanian, B., Balu, A., Hegde, C., Krishnamurthy, A. (2023). Towards foundational AI models for additive manufacturing: Language models for G-code debugging, manipulation, and comprehension, *ArXiv,* doi: 10.48550/arXiv.2309.02465.
[15] Thirunavukarasu, A.J., Ting, D.S.J., Elangovan, K., Gutierrez, L., Tan, T.F., Ting, D.S.W. (2023). Large language models in medicine, *Nature Medicine,* Vol. 29, 1930-1940, doi: 10.1038/s41591-023-02448-8.
[16] Dai, W., Lin, J., Jin, H., Li, T., Tsai, Y.-S., Gašević, D., Chen, G. (2023). Can large language models provide feedback to students? A case study on ChatGPT, In: *Proceedings of 2023 IEEE International Conference on Advanced Learning Technologies (ICALT),* Orem, USA, 323-325, doi: 10.1109/ICALT58122.2023.00100.
[17] Chang, Y., Wang, X., Wang, J., Wu, Y., Yang, L., Zhu, K., Chen, H., Yi, X., Wang, C., Wang, Y., Ye, W., Zhang, Y., Chang, Y., Yu, P.S., Yang, Q., Xie, X. (2024). A survey on evaluation of large language models, *ACM Transactions on Intelligent Systems and Technology,* Vol. 15, No. 3, 1-45, doi: 10.1145/3641289.
[18] Devlin, J., Chang, M.-W., Lee, K., Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding, In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,* Minneapolis, USA, 4171-4186, doi: 10.18653/v1/N19-1423.
[19] Bouschery, S.G., Blazevic, V., Piller, F.T. (2023). Augmenting human innovation teams with artificial intelligence: exploring transformer-based language models, *Journal of Product Innovation Management*, Vol. 40, No. 2, 139-153, doi: 10.1111/jpim.12656.
[20] Pearce, K., Zhan, T., Komanduri, A., Zhan, J. (2021). A comparative study of transformer-based language models on extractive question answering, *ArXiv,* doi: 10.48550/arXiv.2110.03142.

[21] Kalla, D., Smith, N. (2023). Study and analysis of Chat GPT and its impact on different fields of study, *International Journal of Innovative Science and Research Technology*, Vol. 8, No. 3, 827-833, doi: 10.5281/zenodo.7767675.

[22] Chitsaart, C., Rianmora, S., Rattana-Areeyagon, M., Namjaiprasert, W. (2014). Automatic generating CNC-code for milling machine, *International Journal of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering*, Vol. 7, No. 12, 2607-2613.

[23] Zhang, Y., Zeng, Q., Mu, G., Yang, Y., Yan, Y., Song, W. , Gong, Y. (2018). A design for a novel open, intelligent and integrated CNC system based on ISO 10303-238 and PMAC, *Tehnički Vjesnik – Technical Gazette*, Vol. 25, No. 2, 470-478, doi: 10.17559/TV-20170419111243.

[24] Gu, Y., Wang, Y., Lin, J., Yuan, X. (2017). Fault location in CNC system software based on the architecture expansion, *Tehnički Vjesnik – Technical Gazette*, Vol. 24, No. 2, 619-625, doi: 10.17559/TV-20160704190047.

Appendix A

Fig. A1 Query for G-code generation and file recognition using ChatGPT-4o model

I have a 3D part for which I have created an ISO G-code. I will give you the description of the part and then provide the code, which I think contains some errors. Find and fix them.

Sure, I'd be happy to help. Please provide the description of the part and the G-code, and I'll do my best to identify and correct any errors.

Part Cube with a size of 150 mm x 150 mm x 150 mm. The cube is symmetrical around its central vertical axis. There are four symmetrically arranged holes on the top of the cube. Each hole has a diameter of 16 mm and is located 25 mm from the edges of the cube at a depth of 50 mm There is a central through hole with a diameter of 12 mm. Surrounding the central through hole on the top is a milled pocket with a diameter of 25 mm and a depth of 25 mm, which can accommodate the head of a bolt or screw.

Great, that's a clear description of the part. Now, please provide the ISO G-code for the part, and I'll take a look to see if there are any errors.

Some guidelines: I want you to read each line carefully and check for possible errors such as tool size, G0 and G1, spindle rotation on/off, tool position, part feature size and tool size, speeds and feeds etc. additionally I would like you to indicate in which lines (number them) there are errors. Do this in detail!

Absolutely, I'll carefully review each line of the G-code and note any potential errors, including issues with tool size, movement commands, spindle rotation, tool position, part feature sizes, speeds, and feeds. Please provide the G-code, and I'll start the review.

```
%
O0001
G17 G21 G94 G90

(FACE_MILL_ZIGZAG , TOOL : UGT0202_031)

N1 G0 G53 Z762.825
N2 T01 M6
N3 G54
N4 G17 G0 G90 X190. Y20.01 S1061
N5 G43 Z3.4 H1
N6 Z0.2
N7 G94 G1 X165. F250.
N8 X-15.
N9 G2 Y56.67 I0. J18.33
N10 G1 X165.
N11 G3 Y93.33 I0. J18.33
N12 G1 X-15.
N13 G2 Y129.99 I0. J18.33
N14 G1 X165.
N15 X190.
N16 G0 Z3.4
N17 M5
N18 G28 G91 Z0.
N19 G28 X0. Y0.
N20 G90 G28

(DRILLING , TOOL : UGT0301_006)

N21 T02 M6
N22 G54
N23 G17 G0 G90 X25. Y125. S1989 M3
N24 G43 Z11. H2
N25 Z3.
N26 G94 G1 Z-54.807 F250.
N27 G0 Z3.
N28 Z11.
N29 X125.
N30 Z3.
N31 G0 Z-54.807
N32 G0 Z3.
N33 Z11.
N34 Y25.
N35 Z3.
N36 G1 Z-54.807
N37 G0 Z3.
N38 Z11.
N39 X25.
N40 Z3.
N41 G1 Z-54.807
N42 G0 Z3.
N43 Z11.
N44 M5
N45 G28 G91 Z0.
N46 G28 X0. Y0.
N47 G90 G28

(DEEP_HOLE_DRILLING , TOOL : UGT0301_025)

N48 T03 M6
N49 G54
N50 G17 G0 G90 X75. Y75.
N51 G43 Z11. H3
N52 Z3.
N53 S716000 M3
N54 G94 G1 Z-4.803 F125. M26
N55 G4 X1.
N56 Z-147. F250.
N57 G4 X1.
N58 Z-154.303 F125.
N59 G4 X1.
N60 M5
N61 M9
N62 G0 Z3.
N63 Z11.
N64 G28 G91 Z0.
N65 G28 X0. Y0.
N66 G90 G28

(HOLE_MILLING , TOOL : UGT0201_133)

N67 T04 M6
N68 G54
N69 G17 G0 G90 X75. Y75. S2122 M3
N70 G43 Z11. H4
N71 Z3.
N72 G94 G3 X80. Y75. I2.5 J0. F250.
N73 X80. Y75. Z-0.75 I-5. J0.
N74 X78.346 Y71.284 Z-4. I-5. J0.
N75 X78.346 Y71.284 I-3.346 J3.716
N76 X75. Y75. I-1.673 J1.858
N77 X80. Y75. I2.5 J0.
N78 X80. Y75. Z-7.75 I-5. J0.
N79 X78.346 Y71.284 Z-11. I-5. J0.
N80 X78.346 Y71.284 I-3.346 J3.716
N81 X75. Y75. I-1.673 J1.858
N82 X80. Y75. I2.5 J0.
N83 X80. Y75. Z-14.75 I-5. J0.
N84 X78.346 Y71.284 Z-18. I-5. J0.
N85 X78.346 Y71.284 I-3.346 J3.716
N86 X75. Y75. I-1.673 J1.858
N87 X80. Y75. I2.5 J0.
N88 X80. Y75. Z-21.75 I-5. J0.
N89 X78.346 Y71.284 Z-25. I-5. J0.
N90 X78.346 Y71.284 I-3.346 J3.716
N91 X75. Y75. I-1.673 J1.858
N92 G0 Z11.
N93 M5
N94 G49
N95 G53 Z762.825
N96 M2
```

Fig. A2 Query for G-code debugging using ChatGPT-3.5 model (lines with error marked in yellow)

Let's go through the G-code line by line:

1. **Line N1:** Looks fine. It's moving the tool to a specific position.
2. **Line N2:** Tool change command. No apparent issues.
3. **Line N3:** Setting the work offset. No problems here.
4. **Line N4:** Moving to a specific position, indicating start of milling process. No issues.
5. **Line N5:** Tool length compensation. Seems fine.
6. **Line N6:** Moving to a safe Z height before starting milling. No errors.
7. **Line N7:** First milling operation, moving in a straight line. No problems.
8. **Line N8:** Moving to a negative X coordinate. Seems correct.
9. **Line N9:** Arc move. Looks good.
10. **Line N10:** Another straight line move. No issues.
11. **Line N11:** Another arc move. Everything appears correct.
12. **Line N12:** Moving to a negative X coordinate again. Seems fine.
13. **Line N13:** Another arc move. Looks good.
14. **Line N14:** Another straight line move. No errors.
15. **Line N15:** Another straight line move. Seems correct.
16. **Line N16:** Moving the tool to a safe Z height. No problems.
17. **Line N17:** Spindle stop command. No issues.
18. **Line N18:** Returning to home position. Correct.
19. **Line N19:** Additional homing command. Seems fine.
20. **Line N20:** Resetting the work offset. No errors.

For the next section, I'll follow the same method, pointing out any potential errors:

21. **Line N21:** Tool change command. All good.
22. **Line N22:** Setting the work offset. No issues.
23. **Line N23:** Moving to a specific position. Correct.
24. **Line N24:** Tool length compensation and setting Z height. Looks good.
25. **Line N25:** Moving to a safe Z height. No problems.
26. **Line N26:** Drilling operation. Everything seems fine.
27. **Line N27 to N43:** Drilling operations for the remaining holes. All look correct.
28. **Line N44:** Spindle stop command. No issues.
29. **Line N45 to N46:** Returning to home position. Seems correct.
30. **Line N47:** Resetting the work offset. Looks fine.

Now, let's move on to the next section:

31. **Line N48:** Tool change command. All good.
32. **Line N49:** Setting the work offset. Correct.
33. **Line N50:** Moving to a specific position. No issues.
34. **Line N51:** Tool length compensation and setting Z height. Looks good.
35. **Line N52:** Setting spindle speed and initial Z movement. Seems fine.
36. **Line N53 to N60:** Deep hole drilling operations. All look correct.
37. **Line N61:** Tool stop command. No issues.
38. **Line N62 to N66:** Returning to home position. Correct.

Finally, let's check the last section:

39. **Line N67:** Tool change command. All good.
40. **Line N68:** Setting the work offset. Correct.
41. **Line N69:** Moving to a specific position. No issues.
42. **Line N70:** Tool length compensation and setting Z height. Seems fine.
43. **Line N71:** Setting spindle speed and initial Z movement. Looks good.
44. **Line N72 to N91:** Hole milling operations. All seem correct.
45. **Line N92:** Moving the tool to a safe Z height. No problems.
46. **Line N93:** Tool stop command. Correct.
47. **Line N94:** Cancelling tool length compensation. No errors.
48. **Line N95:** Moving the tool to a safe Z height. Looks good.
49. **Line N96:** Program end command. No issues.

After carefully reviewing each line, I couldn't find any apparent errors. The G-code seems well-written and should work as expected. If you have any specific concerns or questions about particular lines, feel free to ask!

Fig. A3 ChatGPT-3.5 model error detection capabilities (problematic part highlighted in red)

Certainly! Here's the complete explanation of each line in the provided ISO G code:

1. `%`: Program start marker.
2. `O0001`: Program number or name.
3. `G17 G21 G94 G90`: Set modes for the CNC machine:
   - `G17`: Select XY plane.
   - `G21`: Set units to millimeters.
   - `G94`: Set feed rate to units per minute.
   - `G90`: Set absolute programming mode.
4. `(FACE_MILL_ZIGZAG , TOOL : UGT0202_031)`: Comment indicating the type of operation ("FACE_MILL_ZIGZAG") and tool used ("TOOL : UGT0202_031").
5. `N1`: Line number.
6. `G0 G53 Z762.825`: Rapid positioning move.
7. `T01 M6`: Tool change command.
8. `G54`: Select coordinate system G54.
9. `G17 G0 G90 X190. Y20.01 S1061 M3`: Rapid move to starting position with specified spindle speed and direction.
10. `G43 Z3.4 H1`: Tool length compensation.
11. `Z0.2`: Move to specified Z position.
12. `G94 G1 X165. F250.`: Linear interpolation move at specified feed rate.
13. `N15 X190.`: Move to specified X position.
14. `X190.`: Repeat previous X-coordinate move (possible redundancy).
15. `G0 Z3.4`: Rapid move to specified Z position.
16. `M5`: Stop spindle rotation.
17. `G28 G91 Z0.`: Return to home position in Z-axis.
18. `G28 X0. Y0.`: Return to home position in XY plane.
19. `G90 G28`: Return to machine zero point.
20. `N21`: Line number.
21. `T02 M6`: Tool change command.
22. `G54`: Select coordinate system G54.
23. `G17 G0 G90 X25. Y125. S1989 M3`: Rapid move to starting position with specified spindle speed and direction.
24. `G43 Z11. H2`: Tool length compensation.
25. `Z3.`: Move to specified Z position.
26. `...` (Similar drilling commands follow).
27. `N48`: Line number.
28. `T03 M6`: Tool change command.
29. `G54`: Select coordinate system G54.
30. `G17 G0 G90 X75. Y75.`: Rapid move to starting position.
31. `G43 Z11. H3`: Tool length compensatior
32. `...` (Similar deep hole drilling commands follow).
33. `N67`: Line number.
34. `T04 M6`: Tool change command.
35. `G54`: Select coordinate system G54.
36. `G17 G0 G90 X75. Y75. S2122 M3`: Rapid move to starting position with specified spindle speed and direction.
37. `G43 Z11. H4`: Tool length compensation.
38. `...` (Similar hole milling commands follow).
39. `N95`: Line number.
40. `M2`: Program end marker.

Each line represents a specific command or action for the CNC machine, contributing to the machining process outlined in the program. Let me know if you need further clarification on any specific aspect!

Fig. A4 ChatGPT-3.5 model G-code explanation

G-code Line	Explanation
`%`	Program start.
`O0001`	Program number 0001.
`G17 G21 G94 G90`	Select XY plane (G17), set units to millimeters (G21), set feed rate to units per minute (G94), set absolute positioning mode (G90).
`(FACE_MILL_ZIGZAG , TOOL : UGT0202_031)`	Comment indicating face milling operation with specified tool.
`N1 G0 G53 Z762.825`	Rapid move (G0) in machine coordinate system (G53) to Z=762.825 mm.
`N2 T01 M6`	Select tool 1 (T01) and tool change (M6).
`N3 G54`	Select work coordinate system 1 (G54).
`N4 G17 G0 G90 X190. Y20.01 S1061 M3`	Ensure XY plane selection (G17), rapid move (G0), absolute positioning (G90) to X=190, Y=20.01, set spindle speed to 1061 RPM (S1061), spindle on clockwise (M3).
`N5 G43 Z3.4 H1`	Apply tool length offset (G43), move Z-axis to 3.4 mm, tool length offset number 1 (H1).
`N6 Z0.2`	Move Z-axis to 0.2 mm.
`N7 G94 G1 X165. F250.`	Feed rate per minute (G94), linear interpolation (G1) to X=165, set feed rate to 250 mm/min (F250).
`N8 X-15.`	Linear move to X=-15 mm.
`N9 G2 Y56.67 I0. J18.33`	Circular interpolation, clockwise (G2) to Y=56.67, center of arc I=0, J=18.33.
`N10 G1 X165.`	Linear interpolation (G1) to X=165 mm.
`N11 G3 Y93.33 I0. J18.33`	Circular interpolation, counterclockwise (G3) to Y=93.33, center of arc I=0, J=18.33.
`N12 G1 X-15.`	Linear interpolation (G1) to X=-15 mm.
`N13 G2 Y129.99 I0. J18.33`	Circular interpolation, clockwise (G2) to Y=129.99, center of arc I=0, J=18.33.
`N14 G1 X165.`	Linear interpolation (G1) to X=165 mm.
`N15 X190.`	Linear move to X=190 mm.
`N16 G0 Z3.4`	Rapid move (G0) to Z=3.4 mm.
`N17 M5`	Spindle stop (M5).
`N18 G28 G91 Z0.`	Return to machine home (G28), incremental positioning (G91), move Z-axis to 0 mm.
`N19 G28 X0. Y0.`	Return to machine home (G28), move X and Y axes to 0 mm.
`N20 G90 G28`	Absolute positioning (G90), return to machine home (G28).
`(DRILLING , TOOL : UGT0301_006)`	Comment indicating drilling operation with specified tool.
`N21 T02 M6`	Select tool 2 (T02) and tool change (M6).
`N22 G54`	Select work coordinate system 1 (G54).
`N23 G17 G0 G90 X25. Y125. S1989 M3`	Ensure XY plane selection (G17), rapid move (G0), absolute positioning (G90) to X=25, Y=125, set spindle speed to 1989 RPM (S1989), spindle on clockwise (M3).
`N24 G43 Z11. H2`	Apply tool length offset (G43), move Z-axis to 11 mm, tool length offset number 2 (H2).
`N25 Z3.`	Move Z-axis to 3 mm.
`N26 G94 G1 Z-54.807 F250.`	Feed rate per minute (G94), linear interpolation (G1) to Z=-54.807 mm, set feed rate to 250 mm/min (F250).
`N27 G0 Z3.`	Rapid move (G0) to Z=3 mm.
`N28 Z11.`	Move Z-axis to 11 mm.
`N29 X125.`	Move X-axis to 125 mm.
`N30 Z3.`	Move Z-axis to 3 mm.
`N31 G1 Z-54.807`	Linear interpolation (G1) to Z=-54.807 mm.
`N32 G0 Z3.`	Rapid move (G0) to Z=3 mm.
`N33 Z11.`	Move Z-axis to 11 mm.
`N34 Y25.`	Move Y-axis to 25 mm.
`N35 Z3.`	Move Z-axis to 3 mm.
`N36 G1 Z-54.807`	Linear interpolation (G1) to Z=-54.807 mm.
`N37 G0 Z3.`	Rapid move (G0) to Z=3 mm.
`N38 Z11.`	Move Z-axis to 11 mm.
`N39 X25.`	Move X-axis to 25 mm.
`N40 Z3.`	Move Z-axis to 3 mm.
`N41 G1 Z-54.807`	Linear interpolation (G1) to Z=-54.807 mm.
`N42 G0 Z3.`	Rapid move (G0) to Z=3 mm.
`N43 Z11.`	Move Z-axis to 11 mm.
`N44 M5`	Spindle stop (M5).
`N45 G28 G91 Z0.`	Return to machine home (G28), incremental positioning (G91), move Z-axis to 0 mm.
`N46 G28 X0. Y0.`	Return to machine home (G28), move X and Y axes to 0 mm.
`N47 G90 G28`	Absolute positioning (G90), return to machine home (G28).
`(DEEP_HOLE_DRILLING , TOOL : UGT0301_025)`	Comment indicating deep hole drilling operation with specified tool.
`N48 T03 M6`	Select tool 3 (T03) and tool change (M6).
`N49 G54`	Select work coordinate system 1 (G54).
`N50 G17 G0 G90 X75. Y75.`	Ensure XY plane selection (G17), rapid move (G0), absolute positioning
`N51 G43 Z11. H3`	Apply tool length offset (G43), move Z-axis to 11 mm, tool length offset number 3 (H3).
`N52 Z3.`	Move Z-axis to 3 mm.
`N53 S716 M3`	Set spindle speed to 716 RPM (S716), spindle on clockwise (M3).
`N54 G94 G1 Z-4.803 F125. M26`	Feed rate per minute (G94), linear interpolation (G1) to Z=-4.803 mm, set feed rate to 125 mm/min (F125), auxiliary function (M26, e.g., coolant on).
`N55 G4 X1.`	Dwell (pause) for 1 second (G4 X1.).
`N56 Z-147. F250.`	Move Z-axis to -147 mm at feed rate 250 mm/min.
`N57 G4 X1.`	Dwell (pause) for 1 second (G4 X1.).
`N58 Z-154.303 F125.`	Move Z-axis to -154.303 mm at feed rate 125 mm/min.
`N59 G4 X1.`	Dwell (pause) for 1 second (G4 X1.).
`N60 M5`	Spindle stop (M5).
`N61 M9`	Coolant off (M9).
`N62 G0 Z3.`	Rapid move (G0) to Z=3 mm.
`N63 Z11.`	Move Z-axis to 11 mm.
`N64 G28 G91 Z0.`	Return to machine home (G28), incremental positioning (G91), move Z-axis to 0 mm.
`N65 G28 X0. Y0.`	Return to machine home (G28), move X and Y axes to 0 mm.
`N66 G90 G28`	Absolute positioning (G90), return to machine home (G28).
`(HOLE_MILLING , TOOL : UGT0201_133)`	Comment indicating hole milling operation with specified tool.
`N67 T04 M6`	Select tool 4 (T04) and tool change (M6).
`N68 G54`	Select work coordinate system 1 (G54).
`N69 G17 G0 G90 X75. Y75. S2122 M3`	Ensure XY plane selection (G17), rapid move (G0), absolute positioning (G90) to X=75, Y=75, set spindle speed to 2122 RPM (S2122), spindle on clockwise (M3).
`N70 G43 Z11. H4`	Apply tool length offset (G43), move Z-axis to 11 mm, tool length offset number 4 (H4).
`N71 Z3.`	Move Z-axis to 3 mm.
`N72 G94 G3 X80. Y75. I2.5 J0. F250.`	Feed rate per minute (G94), circular interpolation, counterclockwise (G3) to X=80, Y=75, center of arc I=2.5, J=0, set feed rate to 250 mm/min.
`N73 X80. Y75. Z-0.75 I-5. J0.`	Circular move to X=80, Y=75, Z=-0.75, center of arc I=-5, J=0.
`N74 X78.346 Y71.284 Z-4. I-5. J0.`	Circular move to X=78.346, Y=71.284, Z=-4, center of arc I=-5, J=0.
`N75 X78.346 Y71.284 I-3.346 J3.716`	Circular move to X=78.346, Y=71.284, center of arc I=-3.346, J=3.716.
`N76 X75. Y75. I-1.673 J1.858`	Circular move to X=75, Y=75, center of arc I=-1.673, J=1.858.
`N77 X80. Y75. I2.5 J0.`	Circular move to X=80, Y=75, center of arc I=2.5, J=0.
`N78 X80. Y75. Z-7.75 I-5. J0.`	Circular move to X=80, Y=75, Z=-7.75, center of arc I=-5, J=0.
`N79 X78.346 Y71.284 Z-11. I-5. J0.`	Circular move to X=78.346, Y=71.284, Z=-11, center of arc I=-5, J=0.
`N80 X78.346 Y71.284 I-3.346 J3.716`	Circular move to X=78.346, Y=71.284, center of arc I=-3.346, J=3.716.
`N81 X75. Y75. I-1.673 J1.858`	Circular move to X=75, Y=75, center of arc I=-1.673, J=1.858.
`N82 X80. Y75. I2.5 J0.`	Circular move to X=80, Y=75, center of arc I=2.5, J=0.
`N83 X80. Y75. Z-14.75 I-5. J0.`	Circular move to X=80, Y=75, Z=-14.75, center of arc I=-5, J=0.
`N84 X78.346 Y71.284 Z-18. I-5. J0.`	Circular move to X=78.346, Y=71.284, Z=-18, center of arc I=-5, J=0.
`N85 X78.346 Y71.284 I-3.346 J3.716`	Circular move to X=78.346, Y=71.284, center of arc I=-3.346, J=3.716.
`N86 X75. Y75. I-1.673 J1.858`	Circular move to X=75, Y=75, center of arc I=-1.673, J=1.858.
`N87 X80. Y75. I2.5 J0.`	Circular move to X=80, Y=75, center of arc I=2.5, J=0.
`N88 X80. Y75. Z-21.75 I-5. J0.`	Circular move to X=80, Y=75, Z=-21.75, center of arc I=-5, J=0.
`N89 X78.346 Y71.284 Z-25. I-5. J0.`	Circular move to X=78.346, Y=71.284, Z=-25, center of arc I=-5, J=0.
`N90 X78.346 Y71.284 I-3.346 J3.716`	Circular move to X=78.346, Y=71.284, center of arc I=-3.346, J=3.716.
`N91 X75. Y75. I-1.673 J1.858`	Circular move to X=75, Y=75, center of arc I=-1.673, J=1.858.
`N92 G0 Z11.`	Rapid move (G0) to Z=11 mm.
`N93 M5`	Spindle stop (M5).
`N94 G49`	Cancel tool length offset (G49).
`N95 G53 Z762.825`	Move in machine coordinate system (G53) to Z=762.825 mm.
`N96 M2`	End of program (M2).

This should provide a clear and coherent view of each line of the G-code along with its explanation. If you have any further questions or need additional details, feel free to ask!

Fig. A5 ChatGPT-4o model G-code explanation