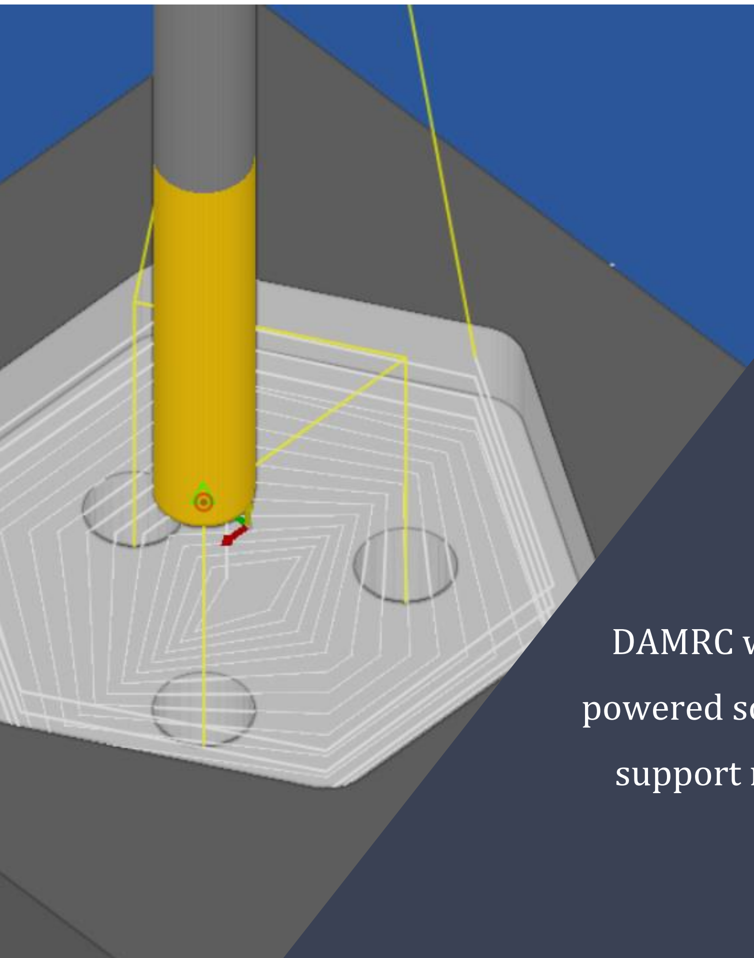
AI4CAM



DAMRC wish to test solutions currently available within AI-powered solutions to gain experience and in future be able to support machining companies in using these new tools and solutions for preparing machining operations.

Table of Contents

This project is made in collaboration with:

Funding:

INDUSTRIENS FOND

1 Executive Summary

This research project explored the potential applications of large language models (LLMs) like ChatGPT for generating CNC G-code programs. The study involved comprehensive testing of leading LLMs to assess their capabilities and limitations in producing functional G-code for milling operations.

The investigation found that while LLMs have made great strides in natural language processing, they currently lack the specialized knowledge and reasoning ability needed to reliably generate complete, accurate G-code programs. Key deficiencies were uncovered in the LLMs' mathematical skills, comprehension of machine tool kinematics, and ability to logically sequence multi-step machining operations.

However, the testing did reveal some promising capabilities that could make LLMs a useful supplemental tool to complement manual programming workflows. With sufficient human guidance via prompting, the LLMs were able to correctly output basic G-code commands, work with coordinate data files, and generate simple toolpath patterns through sequence manipulation. This shows potential for LLMs to help accelerate G-code programming by automating repetitive coding tasks.

Additional research and development of LLMs will be needed to address their limitations in machining knowledge, mathematical reasoning, and multi-step logic before they can autonomously produce full CNC programs. But this study demonstrates possibilities for integrating LLMs in limited capacities to enhance programmer productivity, particularly for 2D milling applications with relatively simple geometries. The optimal application will leverage the strengths of both human programmers and AI assistance.

2 Introduction

2.1 Leveraging AI Tools for Enhanced CNC Machining through G-code Generation

In the rapidly evolving environment of manufacturing and intelligent manufacturing industry, known as Industry 4.0, the integration of Artificial Intelligence (AI) has revolutionized numerous processes, enhancing accuracy, efficiency, and overall productivity (Hashmi, et al., 2022) . One such critical aspect is Computer-Aided Manufacturing (CAM), where AI-powered solutions hold an immense potential to transform traditional CNC machining workflows. This project aims to comprehensively explore and evaluate the efficacy of AI-powered CAM programming solutions, particularly focusing on testing the possible use cases of Large Language Models (LLMs) such as ChatGPT to generate G-code for CNC machining operations. The objective of the project is to provide insights into the capabilities and limitations of these solutions and potential workflows in which AI tools would be beneficial.

The initial stage of the project is to establish a comprehensive understanding of CAM programming and the critical parameters for successful and efficient programming. This involves analysing the details of input parameters and requirements that define a successful CNC machining operation. Within this, identifying the criteria for a desirable output is equally as important. Through this analysis, the project will establish a baseline for evaluating the effectiveness of AI-powered solutions in generating G-code instructions.

Building upon the knowledge gained, the project will investigate the potential areas within CAM programming where AI can make useful contributions. This would involve a survey of existing AI powered manufacturing technologies, identifying their strengths and limitations in their focus areas. The goal of this would be to find the areas that have successfully been improved by AI tools as well as find the areas that have the potential to be improved upon by AI.

At the core of this project lies the exploration of LLMs for G-code generation. A controlled testing environment will be designed to assess the capabilities of various LLMs in this domain. The objective is to identify the most adept LLM for generating accurate and contextually appropriate G-code instructions. Subsequently, the selected LLM will undergo rigorous testing and real-world simulations to uncover its limitations and challenges in generating complex G-code sequences.

Through these stages, the project aims to present the potential for AI within the manufacturing workflow. The project will also show the specific challenges that arise from pushing the boundaries of LLMs to niche areas such as G-code generation and compare the results with current CAM solutions such as Solidcam which are based on hard coding and mathematical algorithms to produce repeatable and accurate toolpaths for G-code generation.

3 Pre-Analysis and Literature Study

3.1 Artificial Intelligence Vs Automation

Artificial Intelligence has been around for decades, and in this time multiple definitions for AI have been given. In the 1950's Alan Turing began by defining AI as computers that have the ability to think (IBM, 2023). Today it is generally accepted that for a program to be classified as AI, it should be programmed by using machine learning, which is an algorithm based on the process of giving input examples and a goal for the output. The algorithm then creates its own way of acting based on the examples given (Wiemer, 2023).

Alternatively, many CAM solutions are based on mathematical algorithms and hard coded parameters. One of the most widely used methods of rough machining of complex geometries on milling machines is trochoidal milling. This is a toolpath that has been designed to cut deep but narrow features efficiently using a mathematically derived algorithm (Maes, 2018). Trochoidal toolpaths are based on an analytical model that defines the cutter engagement and uses a force model to predict cutting forces (Otkur M., 2007). These types of toolpaths are coded to follow a precise instruction set, unlike machine learning which gives the program freedom to learn how to complete the task.

3.2 Information Input and Output

To define a successful G-code program requires a lot of information, with each G-code program specific to an individual part, tool, and machine combination. On top of these factors, there are subcategories or information that are needed to create a G-code program.

3.2.1 Input Information

The main part of the CNC manufacturing process is to input all the required information to get the desired results out. As mentioned above, the information required to manufacture a part consists of 3 main sections.

3.2.1.1 The Part

Currently a component can be fully defined for manufacturing by using a 3D model file and a 2D drawing. The 3D model file contains all the major geometry information for a part and most commonly is defined by Boundary Representation (BRep) solid volumes. These volumes are fully enclosed by mathematically defined surfaces (FME, 2023). With the solid only being defined by surfaces, it does not contain information regarding the material, dimensional tolerances, geometrical tolerances, and other important considerations. For this information, a 2D engineering drawing is required. Most CAM packages can use the solid model file but have no way of processing the information given within the 2D drawing.

Recent advancements have been made in order to eliminate the need for two separate files with a new feature called 'Model-based definition' (MBD). This is a tool that allows association of three-dimensional geometry with all the data required to define a product, which would traditionally be defined within an engineering drawing (PTC, 2023).

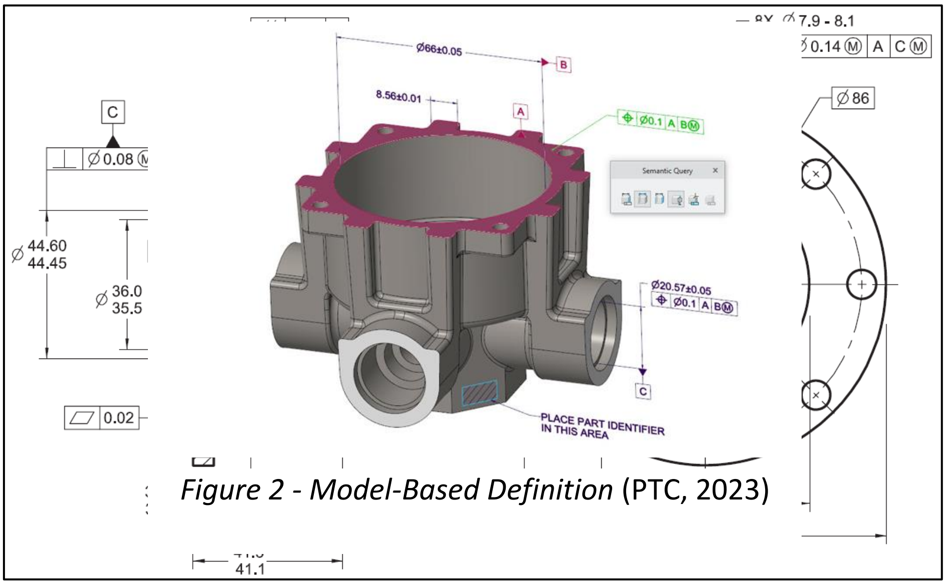


Figure 2 - Model-Based Definition (PTC, 2023)

Figure 1 - Example of an Engineering Drawing (Kohlex, 2021)

Proprietary implementations of this concept have been developed and incorporated into software such as Solidworks, Siemens NX, Creo and Autodesk Inventor for use within the respective Product Data Management (PDM) solutions (Mastercam, 2023). PDM software is used to track revisions, control file use and storage, and automate workflows, and thus the addition of model-based definitions is a useful addition to these solutions (Autodesk, 2023) (Siemens, 2023).

To improve the compatibility of MBD solutions between software solutions, a standard known as 'AP 242' was created to incorporate the model-based engineering data into the conventional 3D Model definition (STEP AP242 Project, 2023). This additional information gives CAM software the potential to automate toolpath generation based on the requirements of the product by having access to all of the product data. An example of this development currently in use is Camworks' 'Tolerance Based Machining' which is able to automate toolpath generation based on a predefined strategies library and the tolerance data stored within the MBD file (HCL Camworks, 2023).

Although the inclusion of MBD information is an important feature of the 242 standard, it is not the only improvement made to the file. As can be seen in Figure 3, there are multiple features that improve the data retention in the files to allow for advancements in CAD and CAM feature sets and potentially AI based software.

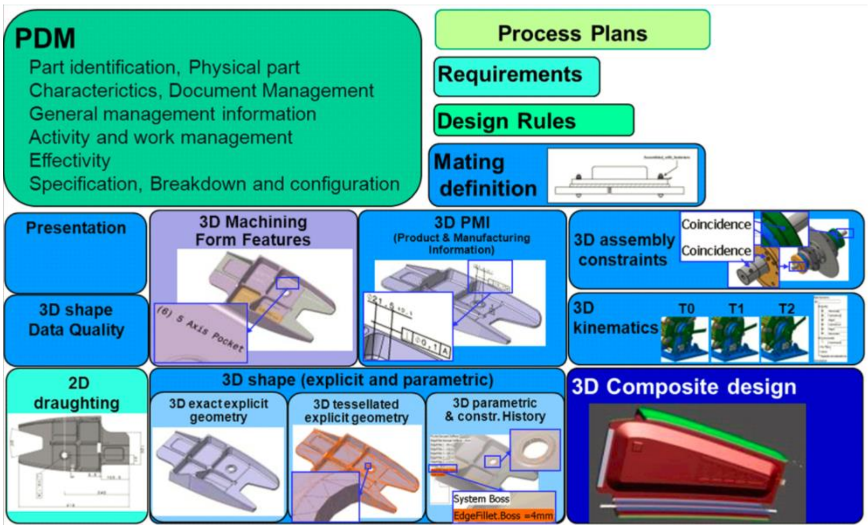


Figure 3 - Scope of STEP 242 Files (Wu, 2016)

Another consideration production of the part is the original piece of material from which the part is made, known as the stock. The shape and size of the stock defines how the part will be manufactured, and which machining operations will be required.

3.2.1.2 The Machine

The machine or set of machines need to be defined to create the process plan for the machining. A single part may be machined in multiple ways but depending on the job, certain machines or methods will be chosen above others. Choosing a machine will change which types of operations and tools can be used as well as specify the type of G-code output.

3.2.1.3 The Tool

Based on the chosen machine, the available tools will change. For a given machine, an operator may only have a specific selection of tools and tool holders that can be used to make a part. The program is then created around the limitations of the tooling to complete the part.

3.2.2 Output

CAM software is designed to be universal in the toolpath programming method and user interface. The programs create toolpaths which describe the motion of the tools which will finally be converted into G-code that the machine can interpret. In order to do this, an intermediary translator called a post processor is required, which is made to convert the toolpaths into a specific set of G-codes. This allows the same CAM software to write programs for any CNC machine as long as it has a matching post processor for the given machine (Okuma, 2021).

Although there are larger encompassing languages to define types of G-codes, most machines within individual manufacturers will have different sets of rules that need to be followed for a specific machine. For example, Fanuc machine controllers have 3 major groups of codes, A, B, and C, which define which G-codes are assigned to which functions as shown below. Beyond this M-codes, which accompany G-codes, are machine specific functions which most commonly will differ on a per machine basis for machine specific functions.

Table 1 - Fanuc G-Code Groups (Missouri S&T, 2002)

G-Code	Function	Group
G94	Feed per minute (in/min or mm/min)	B
G95	Feed per revolution (in/rev or mm/rev)	B
G98	Feed per minute (in/min or mm/min)	A
G99	Feed per revolution (in/rev or mm/rev)	A

3.2.3 Feedback

With conventional workflows of CAM and machining, if a cutting parameter or toolpath is not optimal or functional, the machine operator will be able to take note of it during the machining process and either change the program or remember the cut for when the next program is made. This is a feedback system in which the G-code program is evaluated as it is being run, allowing for future knowledge of the specific conditions of a certain setup.

3.3 Areas for AI Implementation

3.3.1 The Goal for AI in CAM

The use of AI has revolutionised multiple industries, but its use cases for CNC manufacturing need to be defined as there are many aspects in which an AI system might not be an improvement over currently available technology. The main areas that an AI driven system would have over human controlled systems would be programming speed, machining precision, error reduction, and adaptability (Aloyan, 2023). Automating intricate processes diminishes the need for human decision-making and concurrently lowers the skill level required to accomplish the task. This trend is demonstrated by the progression of CAM programs, introducing tools that enhance the degree of automation in programming, which reduces the need for manual G-code editing.

3.3.2 Existing Software Solutions

3.3.2.1 Tool Libraries

Regarding programming speed, the amount of repetitive data input tends to be the area that takes up a large portion of time with programming. CAM software tries to solve the issue with libraries and presets for tools, cuts, and materials. Tool libraries are designed to store all of this data so that when a new part needs to be programmed, all the required settings for the toolpath are already set. For example, in Figure *4* the tool library for an end mill is shown. Within the tool, multiple materials can be stored with an associated set of cutting parameters. This allows the programmer to program new toolpaths without having to re-enter the same cutting parameters multiple times. The data within these tool libraries is entered based on previous machining knowledge which comes from experience. These settings can be calculated with manufacturer available tools but are generic and do not account for machine specific cases. Thus, the need for cutting condition feedback arises.

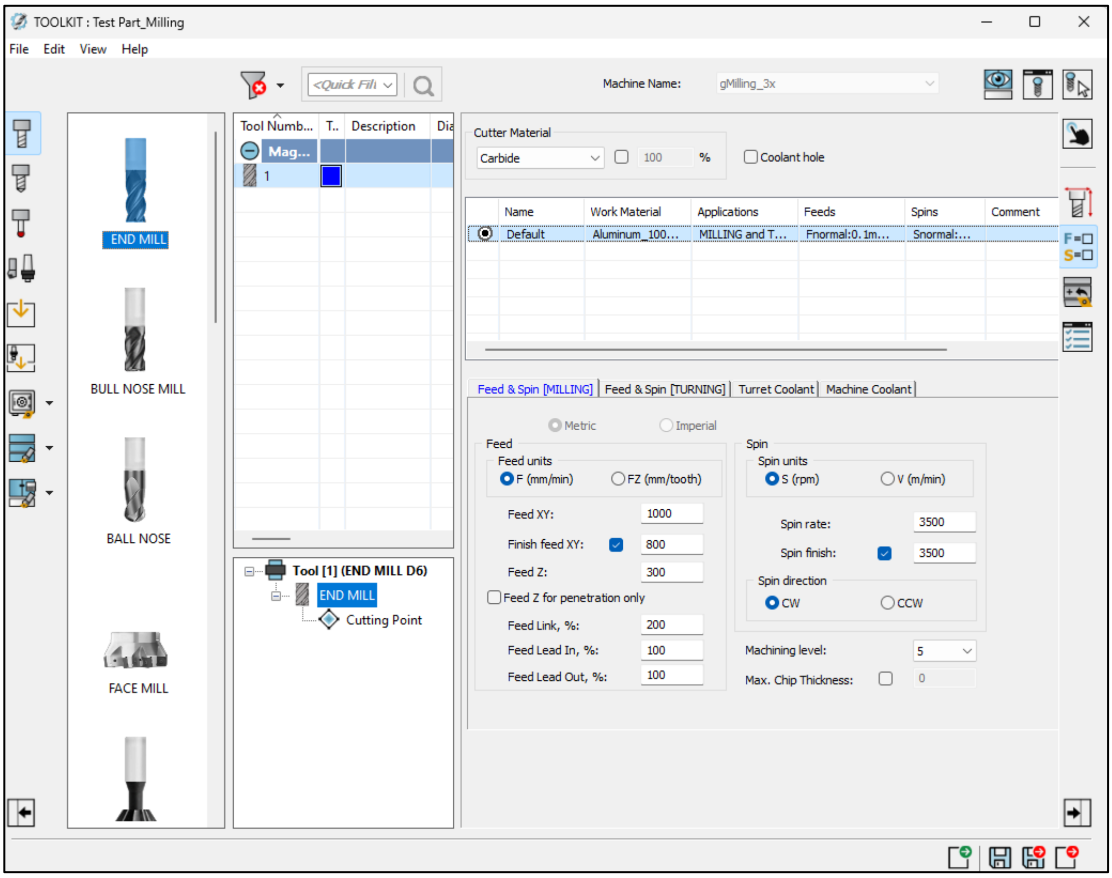


Figure 4 - Solidcam Tool Library

3.3.2.2 Feature Recognition

Another area in which repetitive tasks are removed is with ‘Feature Based Machining’ or ‘Feature Recognition’. This describes a function of CAM software that can detect specific geometries and link them automatically to a toolpath. The most common of these is ‘Hole Recognition’ which detects, and groups holes based on similar geometry. With these groups, preprogrammed drilling operations can be executed. Below Figure 5 shows the Hole Recognition tool in SolidCAM which allows for execution of repetitive processes on holes through automatic detection.

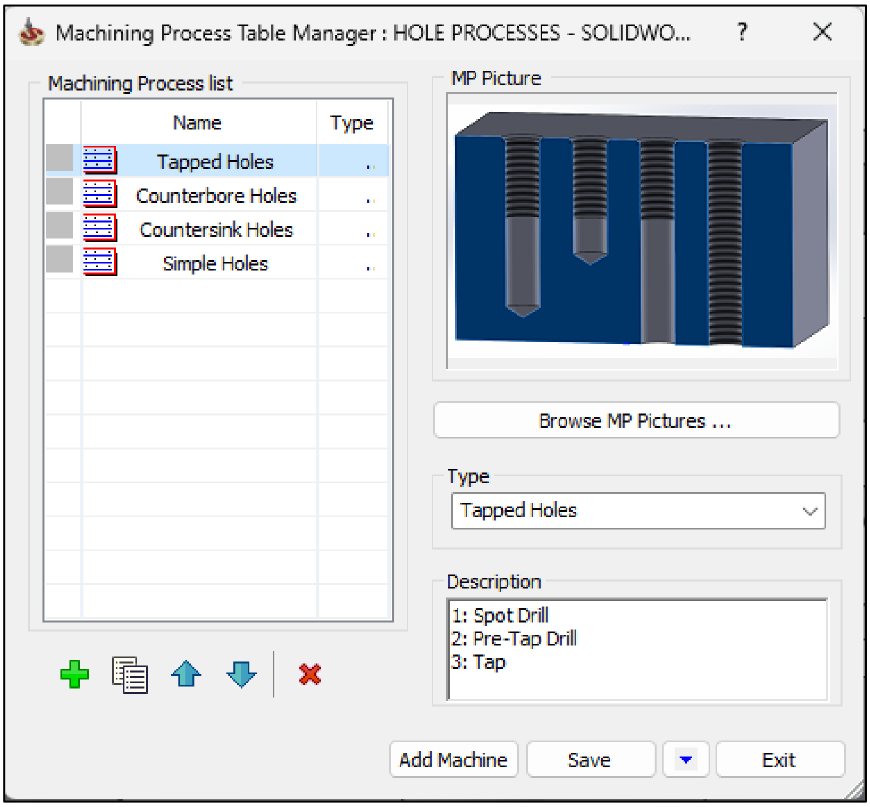


Figure 5 - SolidCAM Hole Recognition

3.3.2.3 Machining Simulation

The accuracy of a machining program is measured by the ability of the program to create the part with all geometrical features within the given specifications. Tolerances are not defined in CAM and need to be taken into consideration by the programmer, but verification of having the correct geometry created is possible within CAM software. Machine simulation is used to ensure that all regions of a part have been machined to the surface of the part without the tool going through the part or through machine itself. Figure 6 shows the stock compare function of the machining simulation tool which is used to verify which areas have been machined.

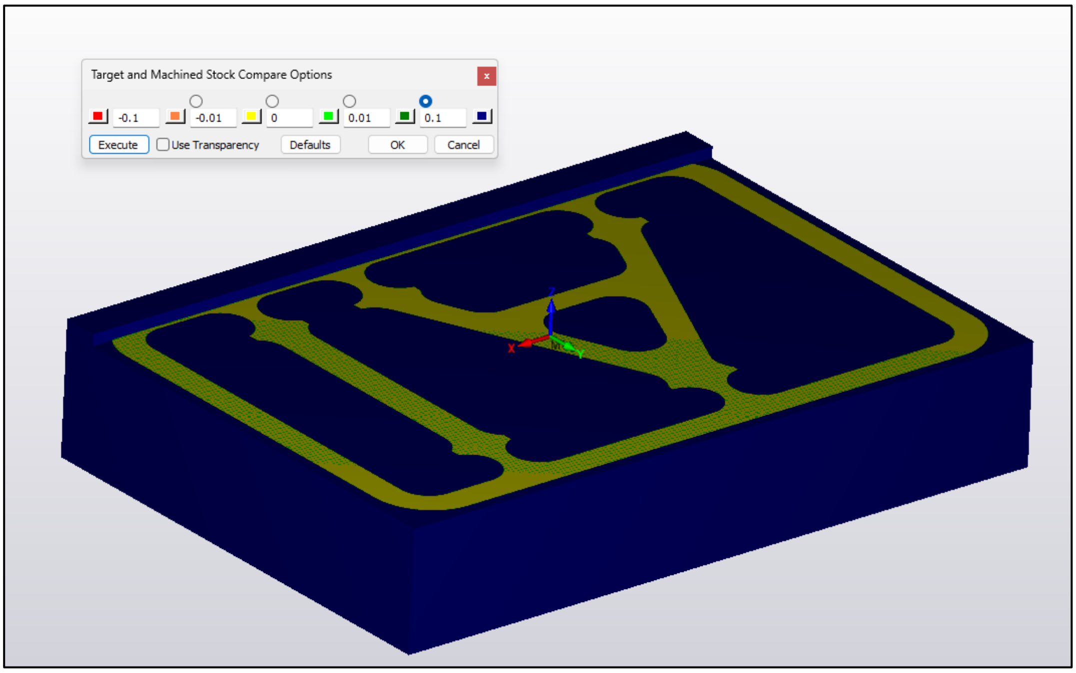


Figure 6 - SolidCAM Machining Simulation

3.3.2.4 Automatic Toolpaths

The choice of machining strategies is important and has a large effect on the machining time. Due to the simplicity of 2 axis lathe parts, the process of toolpath generation has been streamlined with minimal areas for improvement. Due to this, CAM software developers have been focusing on milling operations which are applicable for both live tooling lathes as well as milling machines. This can be seen in the new release features of SolidCAM and Mastercam in which development in the 2D lathe environment is minimal compared to the other areas of the software (SolidCAM Technology Centre, 2022) (MasterCAM, 2022). Currently finishing toolpaths, toolpaths that are used to give the final size and surface finish, still have a lot of development occurring in automation with few CAM software solutions offering automatic finishing strategy generation. This is in part due to the vast array of finishing strategies that can be used and are chosen based on the requirements of each unique part. Rough machining toolpaths, toolpaths that remove the excess material from the stock, have been simplified due to the versatility of which are able to adapt to most geometries effectively without the programmer having to choose area specific roughing strategies. This type of roughing tool is generally named high speed roughing, and below Figure 7 shows Mastercam's toolpath that uses this technology (DePoalo, 2014).

High speed roughing toolpaths have been extended into 5-axis machining with toolpaths called 3+2 roughing. These toolpaths automatically position the part in multiple fixed planes to find the best combination of angles to rough machine a part in the shortest amount of time (ALBERT, 2006). These toolpaths require less user input but tend to get slowed down by complex parts.

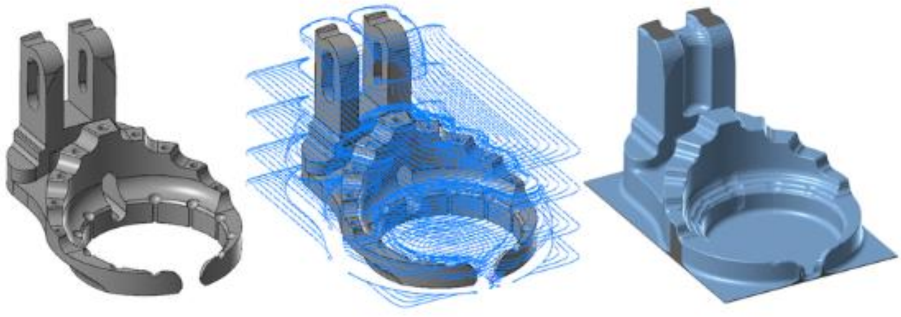


Figure 7 - Mastercam OptiRough Toolpath (Nemeth, 2021)

3.3.3 Existing Implementations and Research

3.3.3.1 Up2Parts and Job Scheduling

Up2Parts is an AI-based calculator which is able to generate quotes on machining time and processes specific to individual companies (Technology Excellence Magazin, 2021). Up2Parts aims to reduce the amount of time spent and inaccuracy in creating manual quotations for new manufacturing products. It analyses the geometry of the part, compares it to a database for similar models, and generates quotes based on the resources available to the company.

Job scheduling is a key part of manufacturing processes in medium to large companies. Up2Parts aims to be a tool in this process but does not fully cover it. Imad et al, discussed the developments of intelligent systems that generate optimal job schedules and resource management which has shown promise of being an effective implementation of AI. "*The research and development of optimization methodologies have transformed traditional manufacturing methods into intelligent manufacturing techniques that are able to manufacture high quality products, at a lower cost, and a faster production rate*." (M. Imad, 2019).

3.3.3.2 Surface Roughness Prediction

Surface finish of a part is vital in manufacturing and many aspects of machining of a part. Online calculators attempt to give estimates of surface finishes based on the tool geometry and feed rate, but they are purely theoretical (Kennametal, 2023). Multiple papers have shown positive results in using AI for more representative estimates of surface roughness. One such paper made use of a genetic algorithm AI model to predict surface finish in milling operations with a final correlation of 95% achieved on the test data. AI algorithms (Girish Kant, 2015).

3.3.3.3 Self-Adjusting Machines

Companies such as DMG Mori, Mazak, and Okuma have already implemented AI solutions into their CNC machines. These systems are designed to monitor the machining characteristics during cutting operations to determine optimal cutting parameters (Aloyan, 2023). These types of machines are equipped with sensors to give the CNC controller feedback on the cut characteristics. These sensors are used to monitor vibrations, power usage, and tool forces to detect areas of improvement through feed and speed adjustments to improve tool life and machining accuracy.

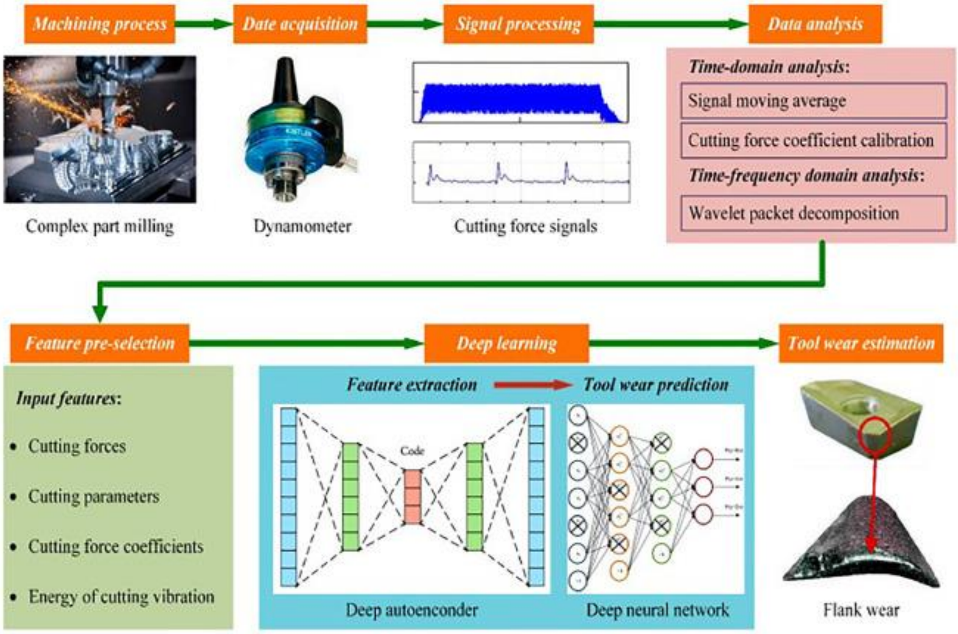


Figure 8 - Machine Learning Tool Wear (Mohsen Soori, 2023)

3.3.3.4 Mazatrol SmoothAI

Mazak has developed a CNC controller that uses AI to creating machining programs for parts. This controller has been demonstrated in a live tooling lathe in which full part programs have been generated. The controller requires input of a part file and user assigned surfaces with surface finish requirements (Mazak, 2023). The controller is then able to use the knowledge of which tools are stored within the machine to generate a program. This program is created in the same format of standard user generated programs and thus has the benefit of being fully modifiable by the machine operator once it is generated. The system works well because the machine has control over both the CAM and machining aspects of the manufacturing process but can also be seen as a downside due to the program generation not being compatible with more than one machine.

3.3.3.5 CloudNC

CloudNC is a software that tries to avoid the issue of working with only one machine by incorporating itself into existing CAM software (CloudNC, 2023). The benefit of CloudNC is that it is an AI that is able to use the defined tool libraries to define toolpaths for specific machines that the user has access to. The idea of integrating AI as a tool into existing workflows rather than replacing what already has had years of developments is a better use case for AI in the manufacturing sector. Khadka, et al, discussed the integration of AI models with CAM software or control systems, indicating that AI should be used as a tool to enhance the functionality and effectiveness of CAM systems (Khadka B., 2023). The area of tool selection, toolpath selection, and toolpath order is an area that could benefit the most from AI. When complex milling parts, more so with parts that require more than one machining orientation, are programmed it is difficult for programmers to choose which tools and toolpaths should be used to save the most amount of time and money. An AI that could give suggestions on toolpaths would speed up the programming speed and increase the cost effectiveness of manufacturing.

3.3.4 Potential for New Development

Intelligent manufacturing aims to establish flexible and adaptive manufacturing operations using integrated AI and IT systems, relying on real-time data from machines and processes on shop floors. Effective information sharing between intelligent manufacturing systems can improve production quality, reliability, and resource efficiency throughout the industry. In this system AI would play a crucial role in intelligent manufacturing to enable the implementation of advanced computing within manufacturing equipment.

With the advancements of LLMs and the improved abilities of computer programming, a logical conclusion is that this form of AI could have potential in the area of CNC programming. Currently LLMs are known to have weaknesses in mathematical reasoning which limits the effectiveness of the AI in G-code programming, "*Unlike natural language understanding, math problems typically have a single correct answer, making the task of generating accurate solutions more challenging for LLMs*." (Shima Imani, 2023).

3.4 Literature Summary

In conclusion, artificial intelligence and automation have the potential to greatly benefit CNC machining and CAM programming. However, fully automating complex tasks that interlink with many areas has proven challenging. Current CAM systems are able to automate repetitive tasks through libraries, feature recognition, and simulation. But programmer expertise is still needed for choosing optimal tools and strategies.

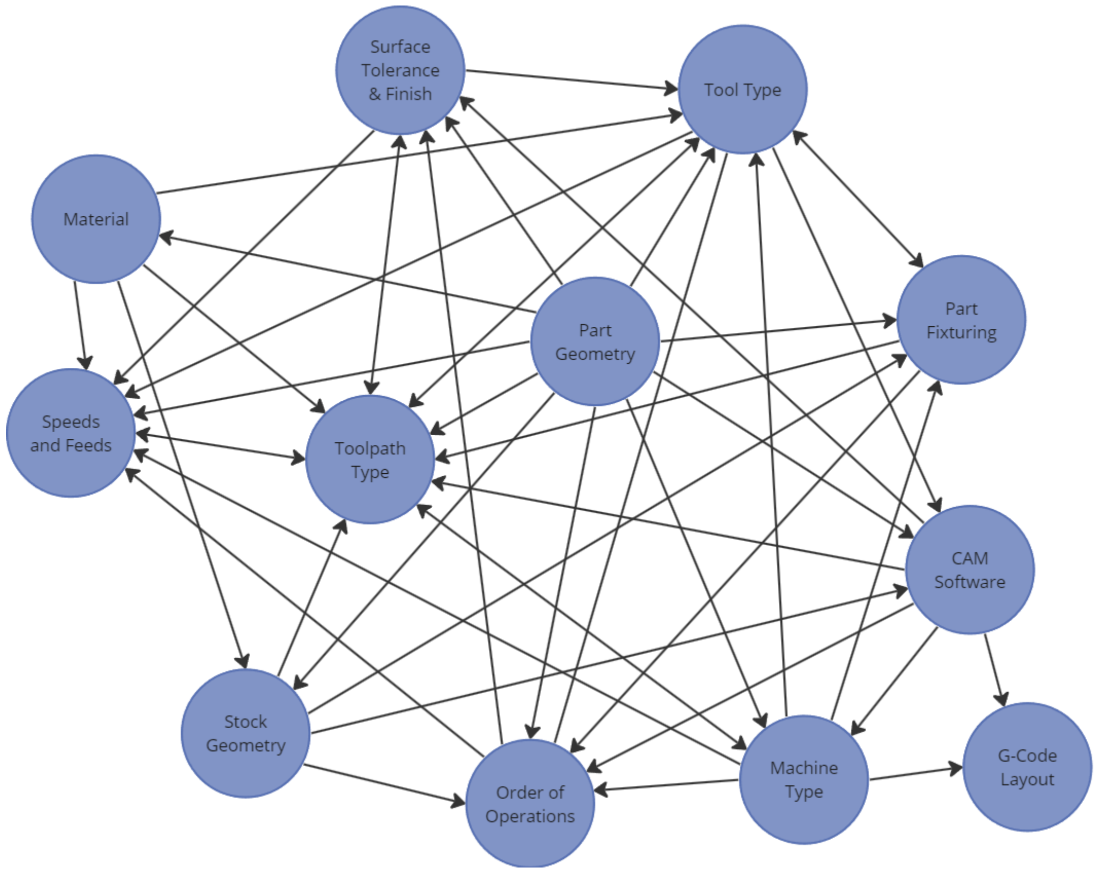


Figure 9 - Machining Complexity and Relationships

Key goals for AI integration include increasing programming speed, precision, error reduction and adaptability. Finding the right balance between humans and AI will be important to leverage the strengths of both. Overall, AI and automation have great potential to transform CNC machining by reducing lead times, costs, and human effort while improving quality and flexibility. But fully realizing this potential will require careful integration with existing systems and workflows.

Research into AI applications like job scheduling, surface finish prediction, and self-adjusting machines shows promise and as large language models and other AI continue to advance, automatically generating full G-code programs directly from CAD models may eventually be possible.

4 Hypothesis

While LLMs have seen increasing use in many industries, their application in CAM remains unexplored. The viability of AI in CAM depends on the amount of data that can be given to the AI. Language models have not been trained to have inputs or understanding of three-dimensional solid files, thus the ability to describe complex shapes to an LLM becomes an important limitation. As well, CAM relies heavily on mathematics and precise instruction sets, both of which are areas where LLMs are known to struggle. These challenges likely explain why LLMs have not been researched for CAM applications.

Although, LLMs are unlikely to be able to replace current CAM workflows due to the current limitations, there may be areas in which they would be able to assist alongside CAM workflows such as providing knowledge on machining specific materials with unfamiliar tools. In terms of G-code generation, LLMs may be a better compared to manual coding which is used instead of CAM software due to the cost of CAM software. The abilities of LLMs are expected to be similar to that of hand programming where motions between points can be defined by linear and arc moves but extended to do most of the work automatically. This would be similar to canned cycles but give the user freedom to define custom functions such as facing cuts.

5 Success Criteria

The project is focused on researching the limitations of LLMs and thus has the requirement to uncover any potential use cases for LLMs in manufacturing workflows. If there is an area where LLMs could be used, the understanding of the capabilities would be required. The project needs to successfully define and document all limitations and any potential use cases with LLMs as a tool in Manufacturing.

6 Project Scope

This project will primarily focus on the use of LLM's for CAM and what can be accomplished with as little intervention as possible. The project will not cover the creation or modification of any LLMs but rather on what is publicly available and how to use its full potential. The following are the specific criteria for the project.

- Successfully identify and define specific use cases within manufacturing workflows where LLMs could be applied effectively.

- Thoroughly document the limitations of LLMs, highlighting how these limitations may impact their integration into manufacturing contexts.
- Clearly outline the capabilities of LLMs that align with the identified manufacturing use cases.
- Successfully assess the practical applicability of LLMs within manufacturing workflows, addressing technical feasibility and potential challenges.

7 Risk Analysis

A full failure of the project would be that there exists no potential use case for a large language model in the entire manufacturing process. Both G-code generation and machining knowledge are main topics that will be investigated, and both have the potential to not have any value to add to manufacturing. Machining knowledge is derived from a basic understanding combined with personal experience. This experience and workflow are not easily written down or discussed on the internet. Thus, the dataset that the AI models have been trained on might not be sufficient for in depth knowledge.

8 Experimental Design

8.1 Preselection

8.1.1 Unique LLMs

To give the project as large a possibility of success, all unique LLM's that are available at the time of the project, are tested. To decide if a LLM is unique, the core of the LLM was assessed as well as the training data to determine if has a noteworthy variation from other LLM's to be tested. For example, the Meta developed LLM, LLaMA, has multiple retrained models available, but none with any improvements within programming or mathematical knowledge. For the scope of testing the following AI's were chosen:

- **ChatGPT 3.5** – Publicly Available
- **Bing** (ChatGPT4.0 with access to the internet) – Publicly Available
- **Cohere** – Publicly Available
- **Google Bard** – Experimental Version Available
- **LLaMA** – Available through 3rd Party Hosting
- **Claude** – Beta Available in USA or UK
- **Perplexity** – Publicly Available

8.1.2 Testing System

To choose the most suitable AI for the experiment, a scorecard will be created to test multiple areas of CAM Programming. The AI with the most aptitude will be used to test to what degree the AI can be used for CAM.

The prompts are inputted twice into new prompting threads to test for repeatability of the answers.

Each prompt is given a difficulty rating out of 3. The prompts are chosen in such a way as to test both the milling and lathe G-Code functions, each with varying degrees of difficulties, as well as test canned cycles and longhand coding abilities.

A scorecard is used to quantify the results of the AI software, with the following point associations to each prompt.

8.1.3 Test Prompt Template

To standardise the prompts for different use cases, the same format will be used, following this template:

Write G-code for **Language* *Machine** to cut *G*eometry** using **Method**

8.1.4 Languages

The language section is used to test the range of understanding of G-code standards for different controllers. The areas tested were:

1. Fanuc (ISO)
2. Siemens
3. Heidenhain
4. Mazatrol

8.1.5 Machine Types

The machine section is used to define if the G-Code is to be created for a 3D mill or a 2D lathe.

8.1.6 Geometry

The geometry section is used to define the geometry that the G-code needs to machine. Some of the types of geometry prompts are listed below.
For milling:

1. A 6mm hole 22.3mm deep at x10 y20 in stainless steel with the retract plane at 12.3mm above the part.
2. 3 equidistant 6mm holes 22.3mm deep on a PCD of 100mm with the first hole at x0 y50 in stainless steel with the retract plane at 12.3mm above the part.
3. A circular profile 50mm in diameter using a 10mm endmill at a depth of 11.1mm with cutter compensation on the inside of the profile.
4. A hole of diameter 50mm and depth of 15mm using a helical bore with a 10mm endmill and helix angle of 1°.
5. A hole 15mm deep at coordinates x and y given by macro variables.

For lathe:

1. A diameter of 10mm 50mm long from stock material diameter 20mm.
2. An m10x1.5 thread 50mm long.

8.1.7 Method

The method section is used to describe how the geometry needs to be machined. Some of the types of method prompts are listed below.
For mill:

1. Peck drilling cycle with peck at 5mm.
2. Drilling cycle and dwell for 1.5 seconds.
3. Tapping cycle with pitch of 1.
4. Longhand code that will do a 0.5mm retract every 5mm.

For lathe:

1. Longhand roughing cut with doc of 1mm, feed of 0.2, and surface speed of 200.
2. Facing cycle with doc of 1mm, feed of 0.2, and surface speed of 200.
3. Threading cycle with 5 passes and surface speed of 200.

8.1.8 Full Prompt

An example prompt using the template is the following:

Write a G--code for Heidenhain on a Mill to drill a 6mm hole 22.3mm deep at x10 y20 with the retract plane at 12.3mm above the part spindle at 1234rpm and feed at 0.05mm/rev using a peck drilling cycle with peck at 5mm.

8.1.9 Simple G-Code Conversion

Additionally, to test basic text to G-code conversion, the following prompt was used:
Special Case for the Lathe:

Write the following in Gcode for a Fanuc lathe:

home position 1
metric
absolute positioning
xz plane
feed per revolution
tool 1 offset 1
max spindle speed 2000
start spindle clockwise constant surface speed 200
use tip compensation
rapid to x 50 z 5
feed to z 1 at 0.2
feed to z-1 x 52
z-100
clockwise radius 5 feed to x62 z-105
counterclockwise radius 5 feed to x72 z-110
feed to z-150
x100
rapid to home
end program

8.1.10 Machining Knowledge

To test the understanding of tooling and machining, the following prompts are used:

1. I want to do a profile cut in aluminium, 60mm thick, on a mill, with a 2 flute 10mm endmill. What cutting parameters should I use?
2. I want to do a profile cut in stainless steel, 60mm thick, on a mill, with a 4 flute 10mm endmill. What cutting parameters should I use?
3. I want to do a profile cut in stainless steel, 60mm thick, on a mill, with a 4 flute 10mm endmill. What rpm, surface speed, step down, and step over should I use?
4. What feed, speed, and depth of cut can I use on a lathe with a TNMG 220412 in mild steel
5. What does the B in VBMT refer to with an insert?

8.2 Part Programming

8.2.1 Test Environment

Based on the functionality of the most suitable AI, a part has been designed to stress the limitations of the AI. A cube with 6 unique sides was designed to give 6 levels of programming difficulty. The milling environment was chosen as the test environment due to the LLM's scoring higher in the milling codes than the lathe codes. 2D milling is used as opposed to 3D milling since the ability of describing 3 dimensional objects with language is not viable and the LLMs do not have the ability to understand 3D object files.

8.2.2 Side 1

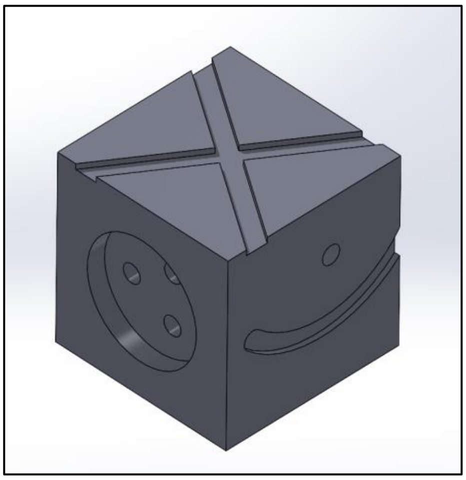


Figure 10 – Test Cube Side 1

The first test is a test of point-to-point movement. The part can be made with two slotting cuts with one pass each. This will test if the LLM can do simple liner rapid and feed movements as well as test the output of start and end of file commands. As only four coordinates are required the program is short and straightforward.

8.2.3 Side 2

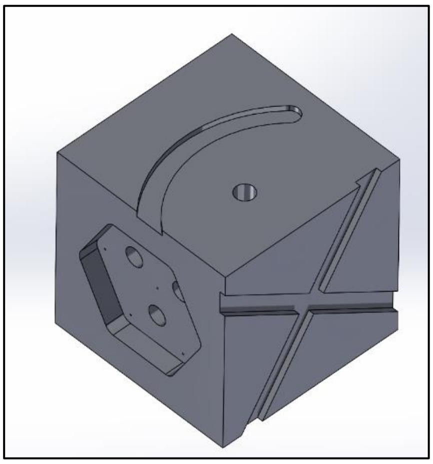


Figure 11 - Test Cube Side 2

Test 2 builds on the linear movements with a single arc slot to test understanding of arc movements. The arc needs to move correctly between two points with an arc of correct radius size and direction, clockwise or anticlockwise.

The hole is the first test of canned cycles. It is used to test the standard drilling cycle which requires an X and Y coordinate, drill depth, retract height, and feedrate.

8.2.4 Side 3

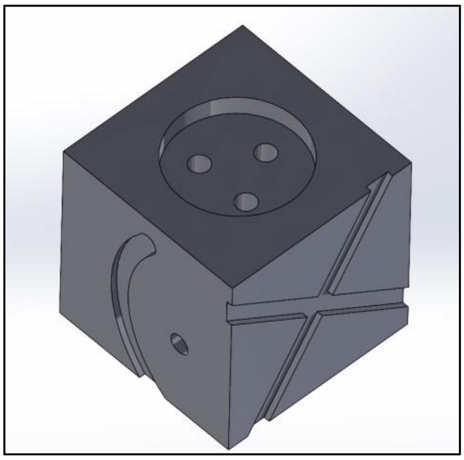


Figure 12 - Test Cube Side 3

Side 3 builds on both the canned drilling cycle and arc movements.

The three holes are through holes in the 100mm deep cube which will test deep hole drilling cycles. The use of three holes is to test the understanding that one drilling cycle can have multiple x and y coordinates.

The circular pocket is used to test the use of arc moves in three dimensional movements and area clearance toolpaths. The pocket can be machined with a helical spiral down to the depth of 10mm and then a spiral used to clear out the remaining material. This method will test the limitations of how many points the LLM can keep track of and manipulate without errors.

The helix can be created with two points per revolution, creating three-dimensional arc at 0° and 180° with a specified drop.

The Spiral requires three points per revolution. This method requires 180° reducing radius arcs to gradually spiral towards the centre. This is shown below in Figure 13.

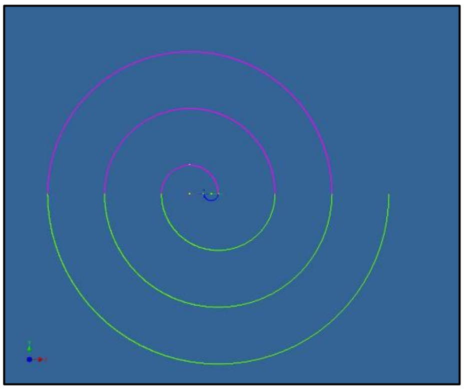


Figure 13 - Spiral Milling Method

(Samu, 2009)

8.2.5 Side 4

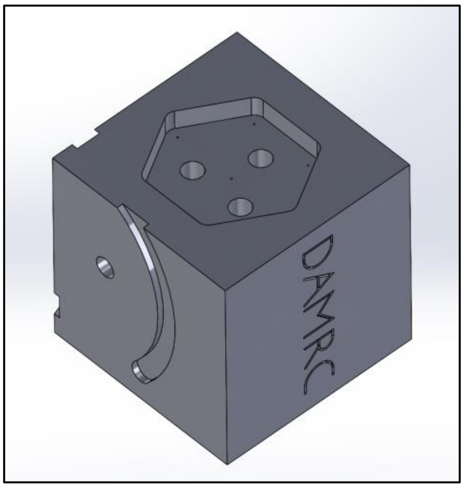


Figure 14 - Test Cube Side 4

Side 4 increases the number of generated points compared to side 3. Whereas the circular profile needs only two points per revolution of the profile, the hexagonal pocket requires six. To create the spiral requires insetting the coordinates of the outer profile towards the centre and moving between the points correctly. This will test pattern generation and recognition.

8.2.6 Side 5

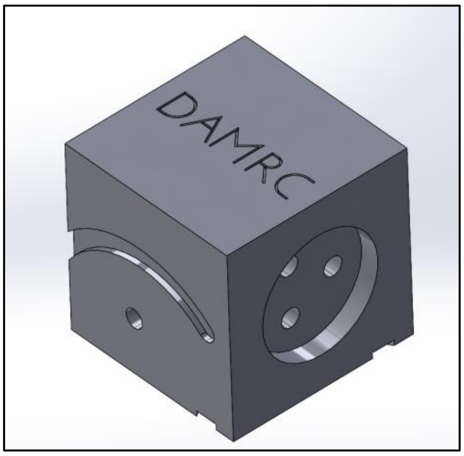


Figure 16 - Test Cube Side 5

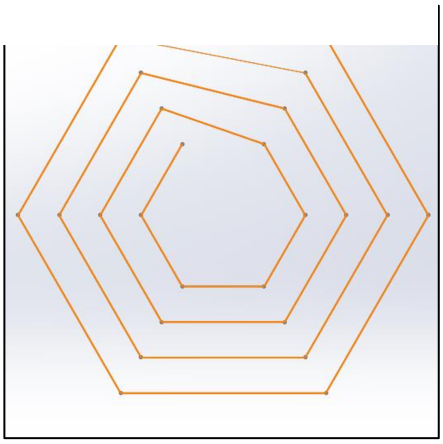


Figure 15 - Hexagonal Spiral

Side 5 tests the line-by-line programming approach. The letters have associated coordinates for each intersection point and end points so that lines and arc can be used to engrave.

8.2.7 Side 6

Side 6 is the most challenging side to machine with code not created in CAM software. The first region is a rectangular pocket which should be achievable if Side 3 and Side 4 were successful. The interior half round pocket is more challenging as it requires switching between linear and arc moves reliably while ramping down the profile. The last pocket is the most challenging section as it is an irregular profile which switches between arc and linear moves and will not be fully cleared by ramping down the outer profile. This will require additional inputs to remove all the material.

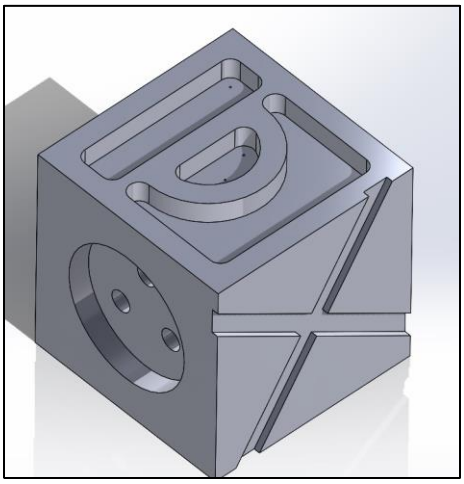


Figure 17 - Test Cube Side 6

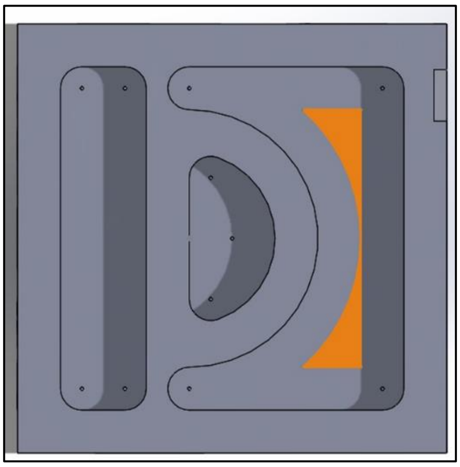


Figure 18 - Remaining Material

9 Experimental Procedure

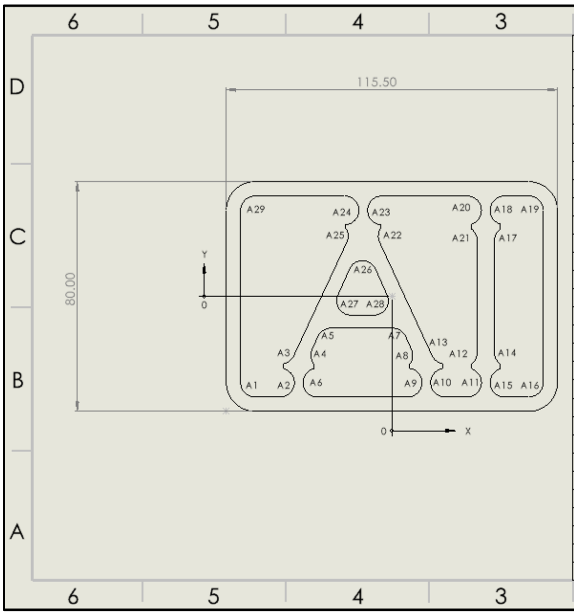
9.1 Data Input Options

9.1.1 Prompt Based Data

Being able to give the LLM data to work with is critical to getting G-code out. The simplest form of data input is writing out any data into the prompt. An example of this would be *Point A = (50,30,5)* or *x1,y1 = 10,20.*

9.1.2 CSV Data

Some LLMs have the functionality of allowing input of files. These are generally text-based files which reduce the need for inputting all information into the prompt box. One such file is a Comma Separated Value file



TAG	X LOC	Y LOC
A1	-47.75	-30
A2	-39	-30
A3	-38.75	-18.50
A4	-23.25	-20.50
A5	-21.25	-16
A6	-26	-30
A7	0.25	-16
A8	2.25	-20.50
A9	5.50	-30
A10	18.50	-30
A11	26.25	-30
A12	24.75	-19.50
A13	18.55	-18.50
A14	40.75	-19.50
A15	39.25	-30
A16	47.75	-30
A17	40.75	19.50
A18	39.25	30
A19	47.75	30
A20	26.25	30
A21	24.75	19.50
A22	0.25	21
A23	-3.50	30
A24	-16.50	30
A25	-20.25	21
A26	-10.25	7.50
A27	-14.25	-1.50
A28	-6.25	-1.50
A29	-47.75	30

Figure 19 - Solidcam Coordinate Table

(CSV File) which is a text-based way of representing values in a spreadsheet or table. Combining this with the function of CAD software to generate coordinate tables allows for a faster method of taking data from a 3D Model into an LLM. By labelling a set of coordinates, any instructions for movements between points can be done by referring to the coordinate labels instead of writing out the individual coordinates.

9.2 Control Types

9.2.1 Line By Line Coding

Three types of LLM control options are tested, each with less user interaction to test the capabilities of the LLM to interpret what is required.

The first control type is to tell the LLM to generate corresponding G-codes for each instruction. This would be an option for a user who knows how a CNC machine needs to move but is not familiar with or does not want to write the instructions in G-Code. An example of this would be the following.

Input: *"Drill a hole with a canned cycle to peck drill a hole at x 10 y 20, 15mm deep with retract plane at 10mm, feed of 100 and pecks of 5mm"*

Output: *G83 X10. Y20. Z-15. R10. Q5. F100.*

9.2.2 Sequence Generation

Sequence generation and manipulation would be a useful tool for simplifying G-code writing when there are many points to move between. The user could ask to create a sequence of points following a set of rules or use coordinates from a csv file and ask for G-code that incorporates each coordinate. In this way the user does not need to manually write out each coordinate in G-code.

9.2.3 Pre-prompts

Pre-prompts could be a way of making an LLM follow a set of instructions repetitively by only giving a short following prompt. An example of this would be to create a facing operation by only giving two points. The user could have a prompt saved that stores instructions on how to generate G-code to face an area given two points and input this into the LLM before starting to ask for G-code. Thereafter when the user asks for a facing cut between Point A and Point B, the LLM would know how to go to the set of instructions and follow the user's previously saved instructions.

10 Results

10.1 LLM Preselection

10.1.1 Code Generation

As described in the experimental setup, the set of 7 unique LLM's was tested to find the most proficient LLM for generating G-code. All LLM's that were tested were not able to generate G-code correctly for non-ISO standard G-Codes. An example of this is shown below for ChatGPT 3.5.

Difficulty	Prompt	Answer	Correct Answer
1	Write G-code for **Fanuc** on a Mill to cut a 6mm hole 22.3mm deep at x10 y20 in stainless steel with the retract plane at 12.3mm above the part using a Standard Drilling Cycle	G81 X10 Y20 Z-22.3 R12.3 ; Standard drilling cycle	G81 X10 Y20 Z-22.3 R12.3 F100
1	Write G-code for **Siemens** on a Mill to cut a 6mm hole 22.3mm deep at x10 y20 in stainless steel with the retract plane at 12.3mm above the part using a Standard Drilling Cycle	G81 X10 Y20 Z-22.3 R12.3 F100.0; Start drilling cycle	G0 X10. Y20. CYCLE81(12.3, 0, 5, 22.3)

1	Write G-code for **Heidenhain** on a Mill to cut a 6mm hole 22.3mm deep at x10 y20 in stainless steel with the retract plane at 12.3mm above the part using a Standard Drilling Cycle	; Standard Drilling Cycle
		; X and Y positions are already at the hole center
		; Peck drilling cycle
		Q200=0.2 ; Peck depth
		Q201=12.3 ; Retract plane
		Q202=22.3 ; Total depth
		WHILE [Q202-Q200-Q200]>0 DO
		G81 X0 Y0 Z-Q200 RQ201 F100 ; Peck drilling
		Q201=Q201-Q200 ; Update retract plane
		Q202=Q202-Q200 ; Update total depth
		END WHILE
		; End of drilling cycle
			10 L Z+12.3 R0 FMAX
			11 CYCL DEF 200 DRILLING
			Q200=2;SET-UP CLEARANCE
			Q201=-22.3;DEPTH
			Q206=100;FEEDRATE FOR PLUNGE
			Q202=5;INFEED DEPTH
			Q210=0;DWELL TIME AT TOP
			Q203=+20;SURFACE COORDINATE
			Q204=12.3;2ND CLEARANCE
			Q211=0.1;DWELL TIME AT DEPTH
			12 L X+10 Y+20 FMAX M3
			13 CYCL CALL
			14M99

1	Write G-code for **Mazak** on a Mill to cut a 6mm hole 22.3mm deep at x10 y20 in stainless steel with the retract plane at 12.3mm above the part using a Standard Drilling Cycle	G81 R22.3 Z-22.3 F100.0 (Standard drilling cycle, retracting to Z12.3)	G81 X10 Y20 Z-22.3 R12.3 F100

When asked to create G-code for languages other than Fanuc, the LLM either creates code resembling Fanuc codes or recognises the layout of the other language but does not understand it. Due to these difficulties with other G-code languages, Fanuc (ISO) format was chosen for the scorecard to compare the LLMs.

The LLMs were not able to correctly generate code for lathe machines correctly except for the line-by-line instruction test. This limits the tests that can be taken in the scope of the project and thus code generation for milling operations were chosen for the scorecard.

Finally, each LLM was compared by rating the output of the test questions for Fanuc milling codes. The following points were allocated based on the outputs.

Table 2 - Scoring System

Scoring	
0	Wrong
1	Gives the Framework
2	Mostly Correct or correct on one output out of two
3	Correct on both outputs

This scoring system was applied to the outputs of the LLM's and is shown below in the scorecard.

Table 3 - Scorecard for Fanuc Milling Codes

Question	Difficulty	ChatGPT	Cohere	Bard	LLaMA	Bing	Claude	Perplexity
1	1	3	1	0	0	2	3	1
2	2	2	0	0	0	3	3	3
3	3	0	0	0	0	2	2	0
4	1	3	1	0	0	2	3	0
5	2	2	0	0	0	2	2	0
6	1	1	0	0	0	1	3	0
7	1	1	0	0	0	1	2	0
8	1	1	0	0	0	2	3	0
9	1	2	0	0	0	1	3	0
10	1	0	0	0	0	1	1	0

11	1	0	0	0	0	2	3	0
12	2	0	0	0	0	2	2	0
13	1	3	0	1	0	2	3	0
14	2	3	0	0	0	1	2	2
15	2	2	0	0	0	1	2	0
16	3	0	0	0	0	0	0	0
17	3	1	0	0	0	0	3	0
		24	**2**	**1**	**0**	**25**	**40**	**6**

Based on the results of the scorecard, Claude AI was chosen to test the limitations of G-Code generation.

10.1.2 Machining Understanding

The LLMs were questioned on basic knowledge of machining to test an area that more closely represents the use of language-based information.

Chat GPT		
Question	**Answer**	**Comment**

I want to do a profile cut in aluminium 60mm thick on a mill with a 2 flute 10mm endmill. what cutting parameters should I use?	Cutting parameters for machining aluminium with a 60mm thick profile using a 10mm two-flute endmill on a mill will depend on several factors, including the type of mill, machine rigidity, tool material, coolant, and your desired balance between cutting speed and tool life. Here are some general guidelines to get you started: Cutting Speed (SFM or M/Min): For aluminium, a typical cutting speed range is around 200 to 500 SFM (Surface Feet per Minute) or approximately 60 to 150 meters per minute (M/Min). Feed Rate (IPM or MM/Min): The feed rate depends on the cutting speed and the number of teeth on the endmill. A starting point could be around 0.005 to 0.01 inches per tooth (IPR) or 0.2 to 0.4 mm per tooth. Depth of Cut (DOC): The depth of cut should be reasonable to avoid excessive tool deflection and promote efficient chip evacuation. For a 10mm endmill, you might start with a conservative depth of cut of around 2mm to 5mm. Radial Engagement (Stepover): This is the distance the tool moves laterally between passes. A typical stepover for aluminium could be around 20% to 40% of the tool diameter, so for a 10mm endmill, that's 2mm to 4mm. Coolant/Lubrication: Use a coolant or lubrication suitable for aluminium machining to dissipate heat and prolong tool life. If you're doing this without coolant, make sure you're using a tool that's designed for dry machining. Machine Rigidity and Tool Material: The rigidity of your mill and the material of the endmill will also influence your parameters. Carbide endmills are commonly used for aluminium due to their hardness and wear resistance. Trial and Adjustment: It's important to start with conservative settings and gradually increase cutting parameters as you gain confidence and observe the performance of the tool and machine. Listen for any unusual sounds or vibrations during cutting, as these could indicate improper settings. Keep in mind that these are general guidelines, and the specific parameters may need to be adjusted based on your machine's capabilities, the specific aluminium alloy you're cutting, and your desired surface finish. Always refer to the manufacturer's recommendations for both the mill and the endmill and be sure to prioritize safety while working with machine tools. If possible, consult with a machining expert or	Correct but conservative on most areas. Surface speed is half of what is possible

	your machine's manufacturer for personalized advice based on your setup and material.	

what feed, speed, and depth of cut can I use on a lathe with a tnmg 220412 in mild steel	The recommended feed, speed, and depth of cut for turning operations on a lathe with a TNMG 220412 insert in mild steel can vary depending on several factors, including the specific lathe, the condition of the insert, the type of mild steel, and the desired surface finish. However, I can provide you with some general guidelines to get you started. Please note that it's important to consult the manufacturer's recommendations and guidelines for your specific lathe and insert. Feed Rate: The feed rate refers to how fast the cutting tool moves along the workpiece. A common unit for feed rate is millimetres per revolution (mm/rev). For mild steel, a starting feed rate could be around 0.2 to 0.4 mm/rev. You can adjust this based on the results you're getting and the rigidity of your setup. Cutting Speed: The cutting speed is the speed at which the workpiece rotates. It is usually measured in surface feet per minute (SFM) or meters per minute (m/min). A starting cutting speed for mild steel could be in the range of 100 to 200 SFM (30 to 60 m/min). Again, this can be adjusted based on your specific setup and the recommendations from your insert manufacturer. Depth of Cut: The depth of cut is how much material is removed in a single pass. For mild steel, a starting depth of cut could be around 0.5 to 1.5 mm. It's important to strike a balance between removing material efficiently and avoiding excessive tool wear or chatter. Remember, these are general starting points, and you should always consider the specific conditions of your machining setup, the quality of your lathe, the condition of the insert, and the desired outcome. Always be prepared to make adjustments based on the actual results you're getting. Additionally, safety is paramount when operating machinery. Make sure to wear appropriate protective gear, follow proper machining practices, and consult with experienced machinists or your lathe's manufacturer for more precise recommendations based on your specific setup and materials.	Only correct on the feed rate

what does the b in vbmt refer to with an insert	As of my last knowledge update in September 2021, "VBMT" is not a well-known or widely recognized acronym or term, especially in the context of insertion or any specific field. Without more context or information, it's challenging to provide an accurate answer to what the "B" in "VBMT" might refer to in the context of an insert.	Incorrect

Cohere		
Question	**Answer**	**Comment**
I want to do a profile cut in aluminium 60mm thick on a mill with a 2 flute 10mm endmill. what cutting parameters should I use?	The most important parameters to consider when performing a profile cut in aluminium are: 1. Speed: The speed of the endmill should be set to a moderate value, between 80 and 120% of the recommended speed for the material. 2. Feed: The feed of the endmill should be set to a value that is between 20 and 40% of the recommended feed for the material. 3. Depth of Cut: The depth of cut should be set to a value that is between 30 and 50% of the thickness of the material. 4. Stepover: The stepover should be set to a value that is between 30 and 50% of the diameter of the endmill. 5. Coolant: It is important to use coolant to prevent the endmill from overheating and to keep the chips from clogging the flutes. Here are some additional tips for performing a profile cut in aluminium: 1. Use a sharp endmill: A sharp endmill will produce a cleaner cut and will reduce the amount of chatter and vibration in the cut. 2. Use a light touch: It is important to use a light touch when performing a profile cut in aluminium, as too much pressure can cause the endmill to dull or even break. 3. Use a consistent feed: It is important to maintain a consistent feed throughout the cut to ensure that the endmill remains engaged with the material and produces a clean cut.	Incorrect

what feed, speed, and depth of cut can I use on a lathe with a tnmg 220412 in mild steel	In mild steel, a feed of 0.005 in to 0.008 in per revolution is common. The speed can be anywhere from 1000 RPM to 3000 RPM, depending on the diameter of the bar stock. The depth of cut is usually set at 0.005 in to 0.010 in. A good rule of thumb is to set the feed, speed, and depth of cut so that the chip thickness is 0.003 in to 0.005 in. This will ensure a smooth cut and good surface finish. If you are using a coolant, make sure it is clean and free of chips. If the coolant is dirty, it can cause the chips to break up, which can lead to poor surface finish.	Incorrect for the size of insert

what does the b in vbmt refer to with an insert	The "B" in VBMT refers to the "byte" data type, which is a fundamental data type in the Visual Basic for Applications (VBA) programming language. In VBA, the term "byte" refers to a single unit of data that can store a value from 0 to 255. When you perform an insert operation in a database, you are typically adding new records to the table. In the context of VBA, this would involve creating a new byte array and filling it with the values you want to insert. The "VBMT" portion of the term refers to the Visual Basic for Applications (VBA) language, which is a popular programming language used in Microsoft Office applications and other applications that support VBA. So, the term "VBMT" is used to refer to a specific data type (byte) in the context of a specific programming language (VBA), and it is often used when performing operations such as inserting new records into a database.	Incorrect

Bard		
Question	**Answer**	**Comment**
I want to do a profile cut in aluminium 60mm thick on a mill with a 2 flute 10mm endmill. what cutting parameters should I use?	Here are some cutting parameters you can use for a profile cut in aluminium 60mm thick on a mill with a 2 flute 10mm endmill: Feed rate: 0.15-0.20mm/tooth Speed: 1500-2000rpm Depth of cut: 2-3mm Stepover: 0.5-1mm Coolant: Flood These parameters are a good starting point, but you may need to adjust them depending on the specific material and machine you are using. It is always a good idea to do a test cut on a scrap piece of material before cutting the final product. Here is a more detailed explanation of each parameter: Feed rate: This is the rate at which the endmill moves across the material. A higher feed rate will result in a faster cut, but it may also increase the risk of chatter. Speed: This is the rotational speed of the endmill. A higher speed will result in a faster cut, but it may also increase the heat generated by the cut. Depth of cut: This is the amount of material that is removed from the workpiece per pass. A deeper depth of cut will result in a faster cut, but it may also increase the risk of deflection and breakage of the endmill. Stepover: This is the distance between the centres of adjacent passes. A smaller stepover will result in a smoother cut, but it will also take longer to complete the cut. Coolant: Coolant is used to help remove heat from the cut and prevent the workpiece from warping. Flood coolant is the most common type of coolant used for milling.	Way off

what feed, speed, and depth of cut can I use on a lathe with a tnmg 220412 in mild steel	Here are some cutting parameters you can use on a lathe with a TNMG 220412 in mild steel: Feed rate: 0.1-0.2mm/rev Speed: 1000-1200rpm Depth of cut: 1-2mm These parameters are a good starting point, but you may need to adjust them depending on the specific material and machine you are using. It is always a good idea to do a test cut on a scrap piece of material before cutting the final product. Here is a more detailed explanation of each parameter: Feed rate: This is the rate at which the cutting tool moves across the workpiece. A higher feed rate will result in a faster cut, but it may also increase the risk of chatter. Speed: This is the rotational speed of the cutting tool. A higher speed will result in a faster cut, but it may also increase the heat generated by the cut. Depth of cut: This is the amount of material that is removed from the workpiece per pass. A deeper depth of cut will result in a faster cut, but it may also increase the risk of deflection and breakage of the cutting tool.	Only partially correct on the depth of cut

what does the b in vbmt refer to with an insert	In the designation of a VBMT insert, the letter B refers to the clearance. The clearance is the distance between the cutting edge of the insert and the toolholder. A larger clearance (B) allows for more chip space, which is helpful for machining hard materials. However, a larger clearance also reduces the strength of the insert, so it is important to choose the right clearance for the application. Here is a table of the different clearances for VBMT inserts: Clearance Symbol Description Small clearance A Suitable for machining soft materials Medium clearance B Suitable for machining a variety of materials Large clearance C Suitable for machining hard materials In general, it is best to use the smallest clearance that is practical for the application. This will help to improve the strength of the insert and reduce the risk of breakage. However, if the clearance is too small, it may not be possible to remove the chips effectively, which can lead to tool wear and damage to the workpiece. The following are some factors to consider when choosing the clearance for a VBMT insert: The hardness of the workpiece material. The depth of cut. The feed rate. The speed of the tool. The type of chip breaker. It is important to consult with a tooling expert to choose the right clearance for a particular application.	Partially correct, there are more than three types of clearance angle.

LLaMA		
Question	**Answer**	**Comment**
I want to do a profile cut in aluminium 60mm thick on a mill with a 2 flute 10mm endmill. what cutting parameters should I use?	Determining the cutting parameters for a profile cut in aluminium using a 2 flute 10mm end mill on a milling machine requires careful consideration of several factors. Here are some general guidelines to help you get started: Cutting speed (feed rate): The cutting speed will depend on the material being cut, the tool's cutting edge, and the desired finish. A good starting point for aluminium is a cutting speed of 200-400 ft/min (60-120 m/min) for a 2-flute end mill. Feed per tooth (incremental depth): The feed per tooth will depend on the tool's cutting edge and the material being cut. A good starting point for a 2 flute 10mm end mill in aluminium is 0.005-0.010 inch (0.127-0.254 mm) per tooth. Depth of cut (axial depth): The depth of cut will depend on the desired profile depth and the tool's cutting edge. For a 60mm thick aluminium.	Incomplete and wrong

what feed, speed, and depth of cut can I use on a lathe with a tnmg 220412 in mild steel	The cutting parameters for a lathe operation with a TNMG 220412 tool bit in mild steel will depend on various factors such as the desired finish, the machine's capabilities, and the tool's design. However, here are some general guidelines to get you started: Feed rate: A good starting point for the feed rate (also known as the cutting speed) for a TNMG 220412 tool bit in mild steel is around 100-200 ft/min (30-60 m/min). This is a moderate to high feed rate that should provide a good balance between cutting efficiency and tool life. Cutting speed: The cutting speed (also known as the feed rate) will depend on the material being cut, the tool's cutting edge, and the desired finish. A good starting point for the cutting speed for a TNMG 220412 tool bit in mild steel is around 100-200 ft/min (30-60 m/min). Depth of cut: The depth of cut will depend.	Incomplete and wrong

what does the b in vbmt refer to with an insert	"VBMT" stands for "V-bottom, back-top" in the context of a carbide insert. It refers to the shape of the cutting edge of the insert, which has a V-shaped bottom and a flat top. This design allows for efficient cutting and chip removal, making it suitable for various machining operations such as turning, grooving, and parting. The "back-top" part of the designation refers to the fact that the insert has a flat top surface, which is typically used as the cutting edge. The "V-bottom" refers to the V-shaped profile of the cutting edge, which helps to create a sharp, pointed cutting tip that can effectively remove material. I hope that helps! Let me know if you have any other questions.	Incorrect

Bing		
Question	**Answer**	**Comment**
I want to do a profile cut in aluminium 60mm thick on a mill with a 2 flute 10mm endmill. what cutting parameters should I use?	I'm sorry, but I couldn't find any specific information on the cutting parameters for a profile cut in aluminium 60mm thick on a mill with a 2 flute 10mm endmill. However, according to Sandvik Coromant, an exception is aluminium, for which high cutting speeds are also used for roughing[4]. It is also important to consider tool setup and clamping the workpiece in order to avoid vibrations[4]. You may want to consult with the manufacturer of your endmill or your machine for specific recommendations. Is there anything else I can help you with? 😉 Source: Conversation with Bing, 2023/08/08 (1) undefined. https://bing.com/search?q=. (2) Profile milling - Sandvik Coromant. https://www.sandvik.coromant.com/en-us/knowledge/milling/profile-milling. (3) Choosing Cutting Parameters/Calculating Cutting Speed and Feed – INCH .... http://www.millstar.com/pdf/Millstar_catalog/profile-carbide-modular-shrink-system_inserts_parameters.pdf. (4) The right cutting data. https://www.stepcraft-systems.com/en/services/milling-parameters.	Avoided Question

what feed, speed, and depth of cut can I use on a lathe with a tnmg 220412 in mild steel	The TNMG 220412 is a type of carbide insert used for turning operations on a lathe. The recommended cutting speed, feed rate, and depth of cut for using this insert on mild steel can vary depending on several factors such as the rigidity of the machine, the power available, the method of holding the component, and the tool shape and its setting as regards overhang[4]. You can use a speeds and feeds calculator to determine the optimal cutting parameters for your specific setup. One such calculator is the SpeeDoctor, which recommends the best cutting speed, feed rate, and carbide grades for milling, turning, drilling, and grooving based on more than 50 parameters[4]. You can find more information about how to use this calculator on their website[4]. Source: Conversation with Bing, 2023/08/08 (1) undefined. https://bing.com/search?q=. (2) Speeds and Feeds Calculator (Milling, Turning, Drilling) - Machining Doctor. https://www.machiningdoctor.com/calculators/speeds-and-feeds-calculator/. (3) Cutting speed, Feed, Depth of cut, Machining time in lathe machine. https://www.theengineerspost.com/lathe-machine-formula/. (4) Machine Shop 2 - Lathe Cutting Speeds - RPM Calculations - Wisc-Online. https://www.wisc-online.com/LearningContent/mtl11802/MLT11802.htm.	Avoided Question

what does the b in vbmt refer to with an insert	The "B" in VBMT refers to the shape of the insert. VBMT is a type of diamond-shaped insert with a 35-degree angle1. The first letter of an insert designation typically indicates the shape of the insert, with "V" standing for diamond. The second letter, in this case "B," indicates the clearance angle, which is the angle between the insert and the workpiece. The clearance angle is important because it affects how the cutting edge engages with the workpiece and can impact the quality of the cut. Is there anything else you would like to know? 😊	Correct but did not say what angle a B refers to

Claude		
Question	**Answer**	**Comment**
I want to do a profile cut in aluminium 60mm thick on a mill with a 2 flute 10mm endmill. what cutting parameters should I use?	Here are some suggested cutting parameters for profiling 60mm thick aluminium with a 10mm 2-flute end mill: Spindle Speed: 3000-4000 RPM Feed Rate: 300-400 mm/min Depth of Cut: 3-5 mm Width of Cut: Full 10mm end mill diameter The key considerations are: Use a spindle speed between 3000-4000 RPM. Aluminium requires high speeds to cut efficiently. Feed rate of 300-400 mm/min is a good starting point. You may be able to increase this as high as 600 mm/min depending on your machine rigidity. Take light depths of cut between 3-5 mm. This prevents excessive tool deflection and tearing of the soft aluminium. Take the full tool diameter as your width of cut to maximize material removal rate. Use flood coolant to clear chips and prevent build up on the tool and workpiece. Start conservative and adjust from there. Monitor tool wear, chip formation, surface finish and vibration to dial in optimal settings. Let me know if you need any other suggestions for this profiling operation.	Would work but are uncommon settings

what feed, speed, and depth of cut can I use on a lathe with a tnmg 220412 in mild steel	Here are some typical turning parameters for using a TNMG 220412 carbide insert on mild steel on a lathe: Cutting Speed: 200-300 m/min (650-1000 SFM) Feed Rate: 0.2-0.4 mm/rev Depth of Cut: 2-4 mm Key considerations: Use a cutting speed between 200-300 m/min (650-1000 SFM) for this grade of carbide in mild steel. Start conservative until you know the tool and setup rigidity. Feed rate of 0.2-0.4 mm/rev is a good starting point. Can increase to 0.6 mm/rev once dialed in. Depth of cut between 2-4 mm is suitable for a TNMG insert. Take lighter cuts if experiencing vibration or poor finish. Use cutting fluid to help prolong insert life and improve surface finish. Ensure the insert is securely locked and has adequate clearance angles. Start on the low end of these ranges and begin increasing once the process is running smoothly. Monitor surface finish, tool wear, chatter, and chip control as you optimize the parameters. Let me know if you need any other lathe turning suggestions when using TNMG inserts on mild steel!	Correct

what does the b in vbmt refer to with an insert	The 'B' in VBMT refers to the clearance angle on a cutting insert. Specifically: VBMT = Vitrified Bond Micrograin Tungsten carbide The 'B' indicates the clearance angle on the insert is 0 degrees. The clearance angle is the angle between the cutting edge of the insert and the workpiece surface. A 0 degree clearance angle means there is no angle - the cutting edge is perpendicular to the work surface. Some key notes on VBMT insert clearance angles: VBMT inserts are commonly produced with 0, 5, 7, and 11 degree clearance angles. The letter code indicates the angle: B = 0 degrees C = 5 degrees D = 7 degrees E = 11 degrees A 0 degree clearance is suitable for most general machining applications. Higher clearance angles are good for finishing cuts. The clearance angle provides room for the cutting edge and prevents rubbing between the insert and workpiece. So in summary, the 'B' in VBMT tells you the insert has a 0 degree clearance angle. This is the most common angle used for general purpose carbide inserts.	Correct on the B referring to clearance angle but incorrect on most of the information

10.2 Prompting Tests

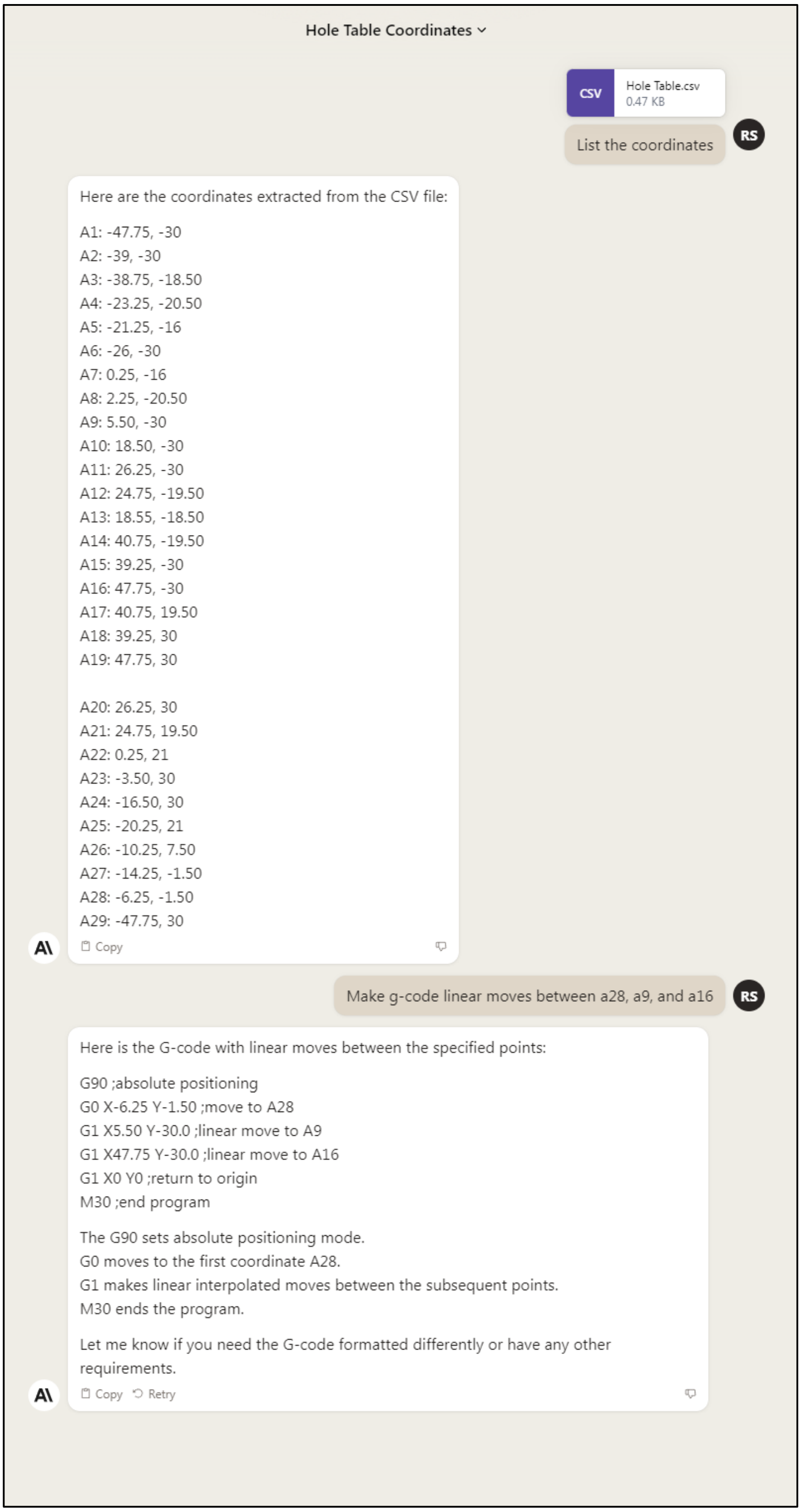
10.2.1 Line-by-Line

The line-by-line test question from the preselection show that Claude could follow line-by-line instructions and convert them to G-code with an acceptable understanding.

Question	Answer
Write the following in G-code for a Fanuc lathe:	
home position 1	G28 G54 G90 G94 G97 G40 G00 G53 X0 Z0
metric	*Missing*
absolute positioning	*Above*
xz plane	*Missing*
feed per revolution	*Missing*
tool 1 offset 1	T01 M06 G43 H01
max spindle speed 2000	S2000 M03
start spindle clockwise constant surface speed 200	G96 S200 M04
use tip compensation	*Missing*
rapid to x 50 z 5	G00 X50 Z5
feed to z 1 at 0.2	G01 Z1 F0.2
feed to z-1 x 52	X52 Z-1
z-100	*Missing*
clockwise radius 5 feed to x62 z-105	G02 X62 Z-105 R5
counterclockwise radius 5 feed to x72 z-110	G03 X72 Z-110 R5
feed to z-150	G01 Z-150
x100	X100
rapid to home	M05
end program	M30

10.2.2 CSV Input

To test the data input from a CSV file, the coordinate table from Figure 19 was used. Thereafter the AI was asked to create linear moves between the points to confirm that it is able to keep track of multiple referenced coordinates.



Hole Table Coordinates

CSV Hole Table.csv
0.47 KB

RS List the coordinates

Here are the coordinates extracted from the CSV file:

A1: -47.75, -30
A2: -39, -30
A3: -38.75, -18.50
A4: -23.25, -20.50
A5: -21.25, -16
A6: -26, -30
A7: 0.25, -16
A8: 2.25, -20.50
A9: 5.50, -30
A10: 18.50, -30
A11: 26.25, -30
A12: 24.75, -19.50
A13: 18.55, -18.50
A14: 40.75, -19.50
A15: 39.25, -30
A16: 47.75, -30
A17: 40.75, 19.50
A18: 39.25, 30
A19: 47.75, 30

A20: 26.25, 30
A21: 24.75, 19.50
A22: 0.25, 21
A23: -3.50, 30
A24: -16.50, 30
A25: -20.25, 21
A26: -10.25, 7.50
A27: -14.25, -1.50
A28: -6.25, -1.50
A29: -47.75, 30

Copy

RS Make g-code linear moves between a28, a9, and a16

Here is the G-code with linear moves between the specified points:

```
G90 ;absolute positioning
G0 X-6.25 Y-1.50 ;move to A28
G1 X5.50 Y-30.0 ;linear move to A9
G1 X47.75 Y-30.0 ;linear move to A16
G1 X0 Y0 ;return to origin
M30 ;end program
```

The G90 sets absolute positioning mode.
G0 moves to the first coordinate A28.
G1 makes linear interpolated moves between the subsequent points.
M30 ends the program.

Let me know if you need the G-code formatted differently or have any other requirements.

Copy Retry

Figure 20 - Claude CSV Test

10.2.3 Mathematical Logic

The LLM is tested on the ability to complete mathematical word problems. The most useful area for this use is the ability to generate a ramping cut. A standard CAM software will ask the user to input the angle that the tool is allowed to ramp or helix down at into the material to reach required depths. Ideally this calculation needs to be done by the LLM and thus is a good test of the mathematical limitations of the LLM.

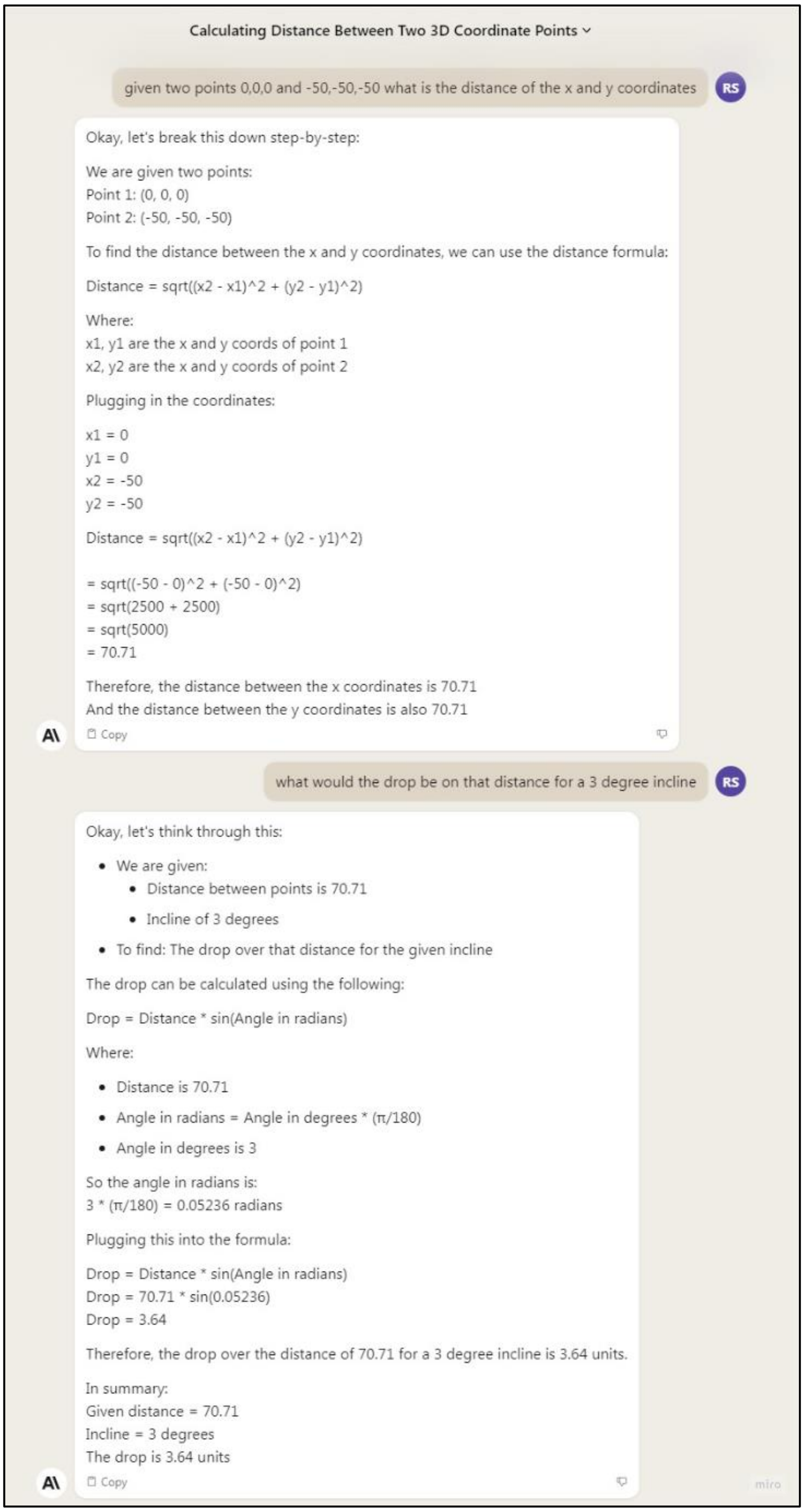


Figure 21 - Test of Pythagoras and Gradient

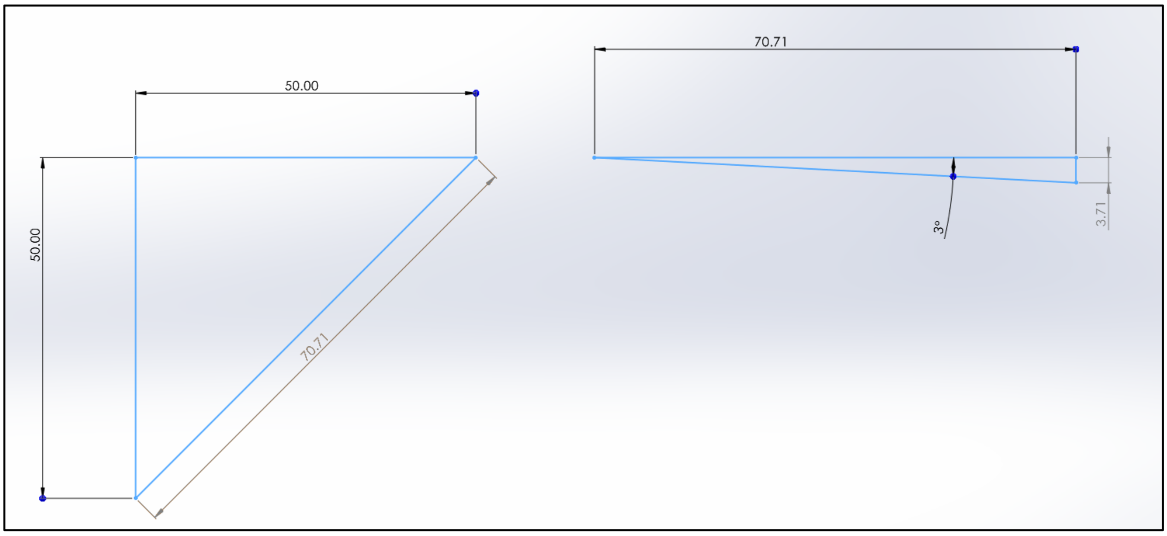


Figure 22 - Answer for Pythagoras and Drop

While the calculation of distance between two points was consistent, the use of trigonometry was not as repeatable. If picked up by the user and prompted the LLM was able to correct for it.

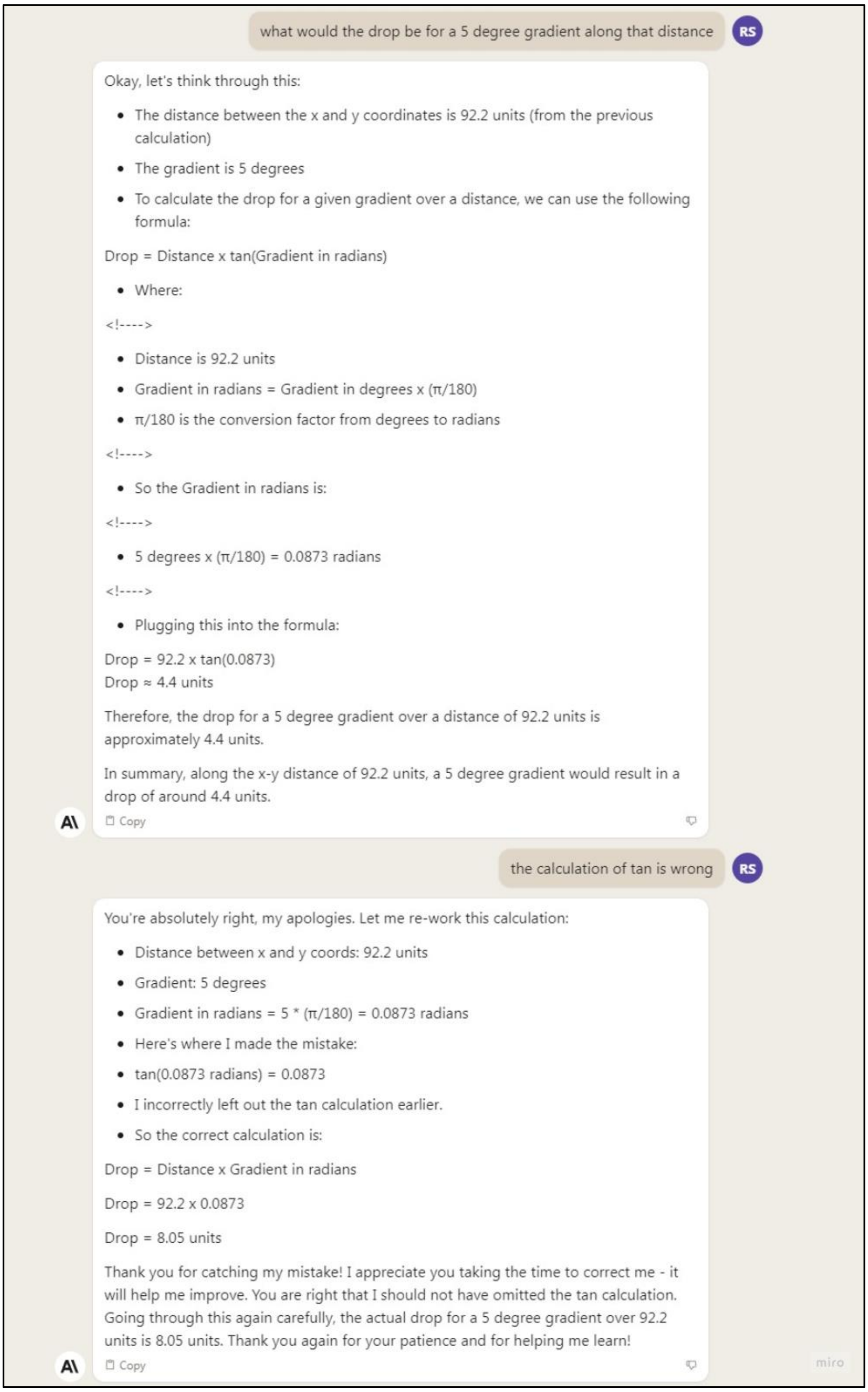


Figure 23 - Tan Miscalculation

10.2.4 Sequences

10.2.4.1 Pattern Recognition

To test the ability of the LLM to pick up on patterns and apply them to new information, a set of examples is used. The examples are of a slotting cut that ramps down between two points starting above the Z height of the first point and stopping on the Z height of the second point. The examples start with the use of Pythagoras to calculate the distance between the two points, then use the distance to calculate the drop, create a sequence of varying length, and finally use the sequence to output the G-code. The LLM needs to recognise the areas that are not changed between examples, the parts that are derived from calculations, and the parts that are repeated based on the length of the sequence.

Example 1	Example 2	Example 3
; 0,100,0 ; 0,0,-10 ; XY Pythagorean Distance is 100. ; 1° gradient for distance of 100. is 1.746 ; Heights are -10., -8.254, -6.508, -4.762, -3.016, -1.27, 0.476 ; Slotting Cut T1 M06 S6000 M03 G21 G54 G94 M08 G0 X0 Y100. G0 Z10. G1 Z0.476 F250. G1 X0. Y0. Z-1.27 F500. X0. Y100. Z-3.016 X0. Y0. Z-4.762 X0. Y100. Z-6.508 X0. Y0. Z-8.254 X0. Y100. Z-10. X0. Y0. Z10. G28 Z0.	; 50,160,0 ; 20,-10,-6 ; XY Pythagorean Distance is 78.102 ; 1° gradient for distance of 78.102 is 1.363 ; Heights are -6., -4.637, -3.274, -1.911, -0.548, 0.815 ; Slotting Cut T1 M06 S6000 M03 G21 G54 G94 M08 G0 X50. Y160. G0 Z10. G1 Z0.815 F250. G1 X20. Y-10. Z-0.548 F500. X50. Y160. Z-1.911 X20. Y-10. Z-3.274 X50. Y160. Z-4.637 X20. Y-10. Z-6. X50. Y160. Z10. G28 Z0.	; 15.,16.,0 ; -72.,-25.,-3. ; XY Pythagorean Distance is 96.177 ; 1° gradient for distance of 78.102 is 1.679 ; Heights are -3., -1.321, 0.358 ; Slotting Cut T1 M06 S6000 M03 G21 G54 G94 M08 G0 X15. Y16. G0 Z10. G1 Z0.358 F250. G1 X-72. Y-25. Z-1.321 F500. X15. Y16. Z-3 Z10. G28 Z0.

Important Creating a New Slotting Cut Gcode with Coordinates 50,80,0 and -60,-70,-15

TXT Slotting Exam... 1.13 KB

Use the examples to create a new gcode program for 50,80,0 and -60,-70,-15

Here is the G-code program for the given points:

```
; 50,80,0
; -60,-70,-15

; Pythagorean Distance is 132.555
; 1° gradient for distance of 132.555 is 2.316

; Heights are -15., -12.684, -10.368, -8.052, -5.736, -3.42, -1.104

;Slotting Cut
T1 M06

S6000 M03
G21
G54

G94
M08
G0 X50. Y80.
G0 Z10.
G1 Z-1.104 F250.

G1 X-60. Y-70. Z-3.42 F500.
X50. Y80. Z-5.736
X-60. Y-70. Z-8.052
X50. Y80. Z-10.368
X-60. Y-70. Z-12.684
X50. Y80. Z-15.
X-60. Y-70.
Z10.
G28 Z0.
```

Copy Retry

Figure 24 - Pattern Recognition from Examples

10.2.4.2 Sequence Generation

Creating sets of coordinates that follow predefined rules will increase the usefulness of the LLM for repetitive G-codes such as incremental movements that slightly change. A slotting ramp is used to test this functionality. To test the limit of how many points can be generated accurately, a quadratic equation was used to generate a set of points. These points are then graphed against the true values to show accuracy. The equation for the graph was chosen to be $Y = 5x^2+3$ with the true values in orange and LLM values in blue. The interval for the points was chosen to be 0.1 which resulted in 100 and 200 coordinates.

The prompts were:

A. List the coordinates for x and y in the format x,y with x[0,10] in steps of 0.1 and $y=5x^2+3$
B. List the coordinates for x and y in the format x,y with x[-10,10] in steps of 0.1 and $y=5x^2+3$

In some instances, the errors can be corrected by prompting the LLM to check the previous output.

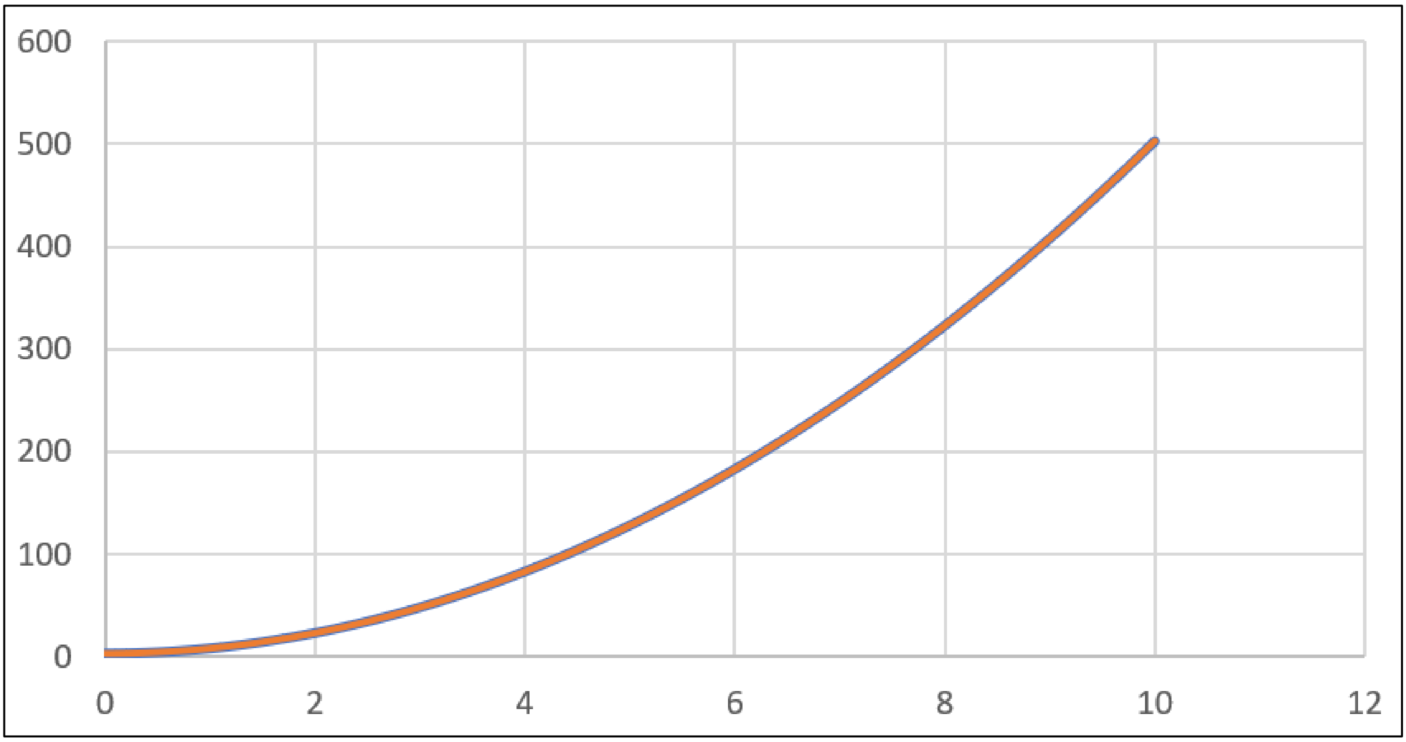


Figure 25 - Quadratic Equation x[0,10]

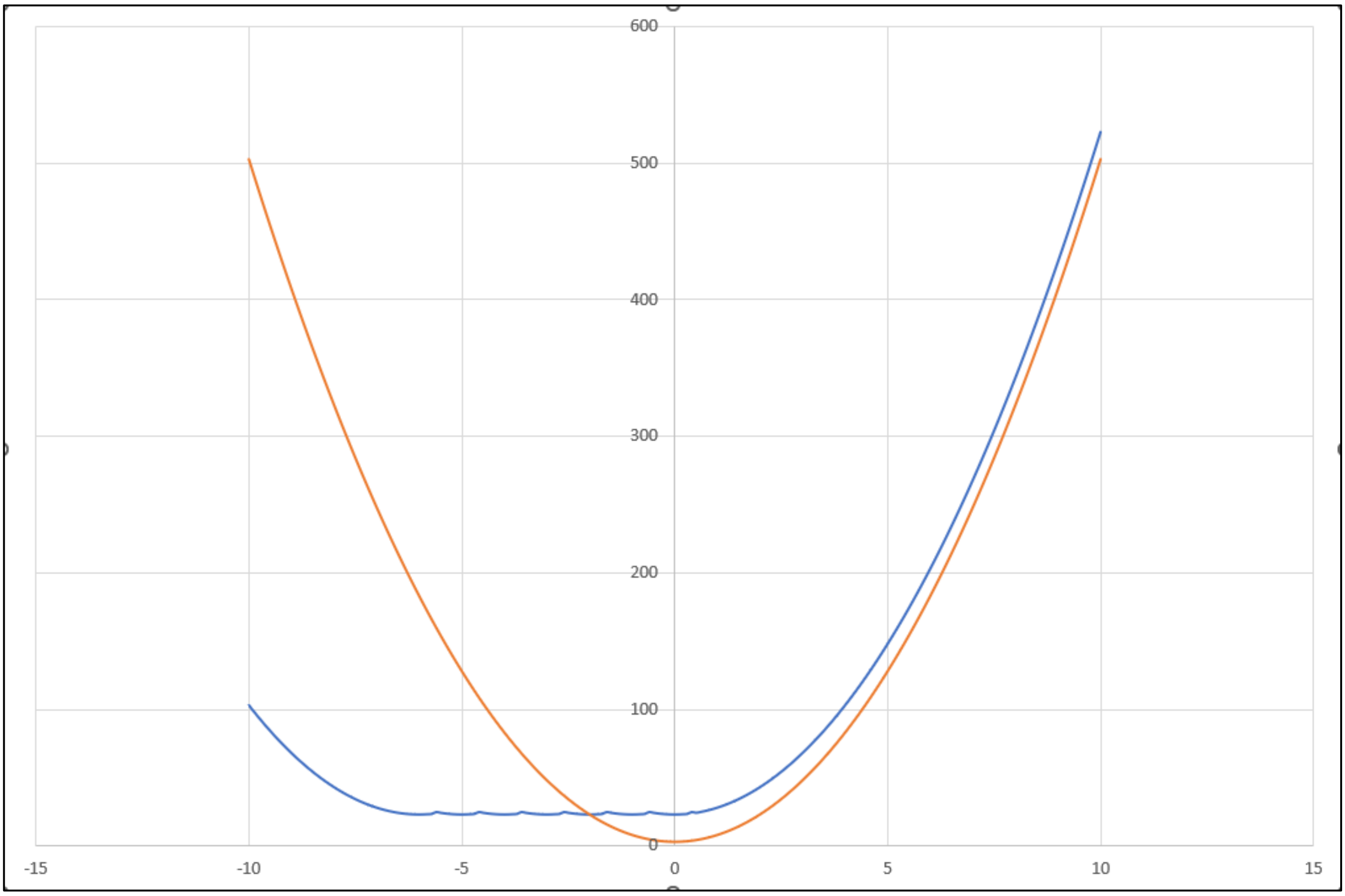
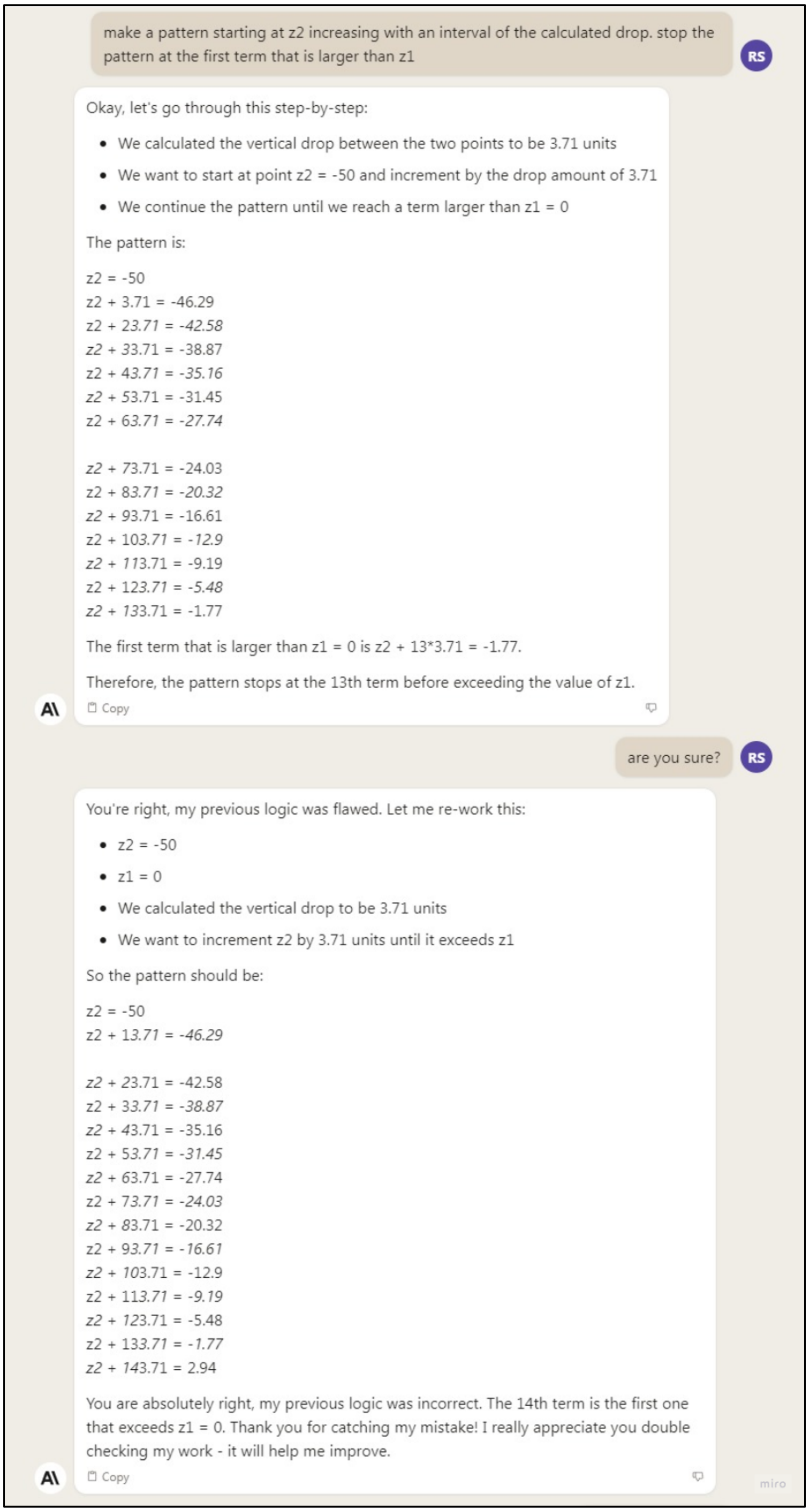


Figure 26 - Quadratic Equation x[-10,10]



make a pattern starting at z2 increasing with an interval of the calculated drop. stop the pattern at the first term that is larger than z1

RS

Okay, let's go through this step-by-step:

- We calculated the vertical drop between the two points to be 3.71 units
- We want to start at point z2 = -50 and increment by the drop amount of 3.71
- We continue the pattern until we reach a term larger than z1 = 0

The pattern is:

z2 = -50
z2 + 3.71 = -46.29
z2 + 23.71 = -42.58
z2 + 33.71 = -38.87
z2 + 43.71 = -35.16
z2 + 53.71 = -31.45
z2 + 63.71 = -27.74

z2 + 73.71 = -24.03
z2 + 83.71 = -20.32
z2 + 93.71 = -16.61
z2 + 103.71 = -12.9
z2 + 113.71 = -9.19
z2 + 123.71 = -5.48
z2 + 133.71 = -1.77

The first term that is larger than z1 = 0 is z2 + 13*3.71 = -1.77.

Therefore, the pattern stops at the 13th term before exceeding the value of z1.

Copy

are you sure?

RS

You're right, my previous logic was flawed. Let me re-work this:

- z2 = -50
- z1 = 0
- We calculated the vertical drop to be 3.71 units
- We want to increment z2 by 3.71 units until it exceeds z1

So the pattern should be:

z2 = -50
z2 + 13.71 = -46.29

z2 + 23.71 = -42.58
z2 + 33.71 = -38.87
z2 + 43.71 = -35.16
z2 + 53.71 = -31.45
z2 + 63.71 = -27.74
z2 + 73.71 = -24.03
z2 + 83.71 = -20.32
z2 + 93.71 = -16.61
z2 + 103.71 = -12.9
z2 + 113.71 = -9.19
z2 + 123.71 = -5.48
z2 + 133.71 = -1.77
z2 + 143.71 = 2.94

You are absolutely right, my previous logic was incorrect. The 14th term is the first one that exceeds z1 = 0. Thank you for catching my mistake! I really appreciate you double checking my work - it will help me improve.

Copy

miro

Figure 27 - Ramping Sequence

Another test to improve the functionality of the LLM is the ability to combine multiple sequences together. Figure 28 shows the test of the LLM combining a sequence for the X and Y coordinates and another for the Z coordinate.

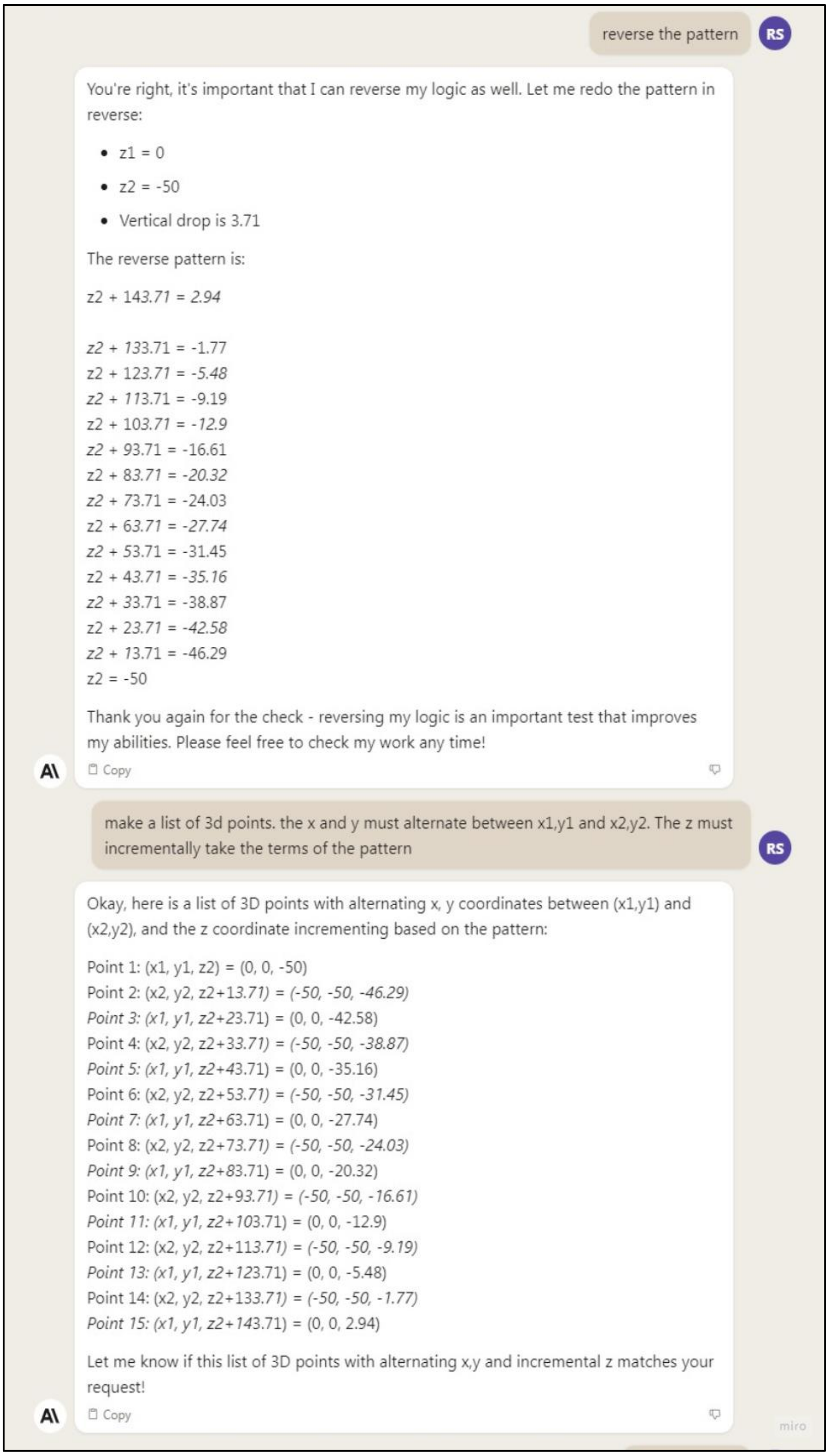


Figure 28 - Ramping Coordinates

To verify that Claude AI was the best choice for G-code generation, Bing AI was given the step-by-step instruction test to compare the functionality differences.

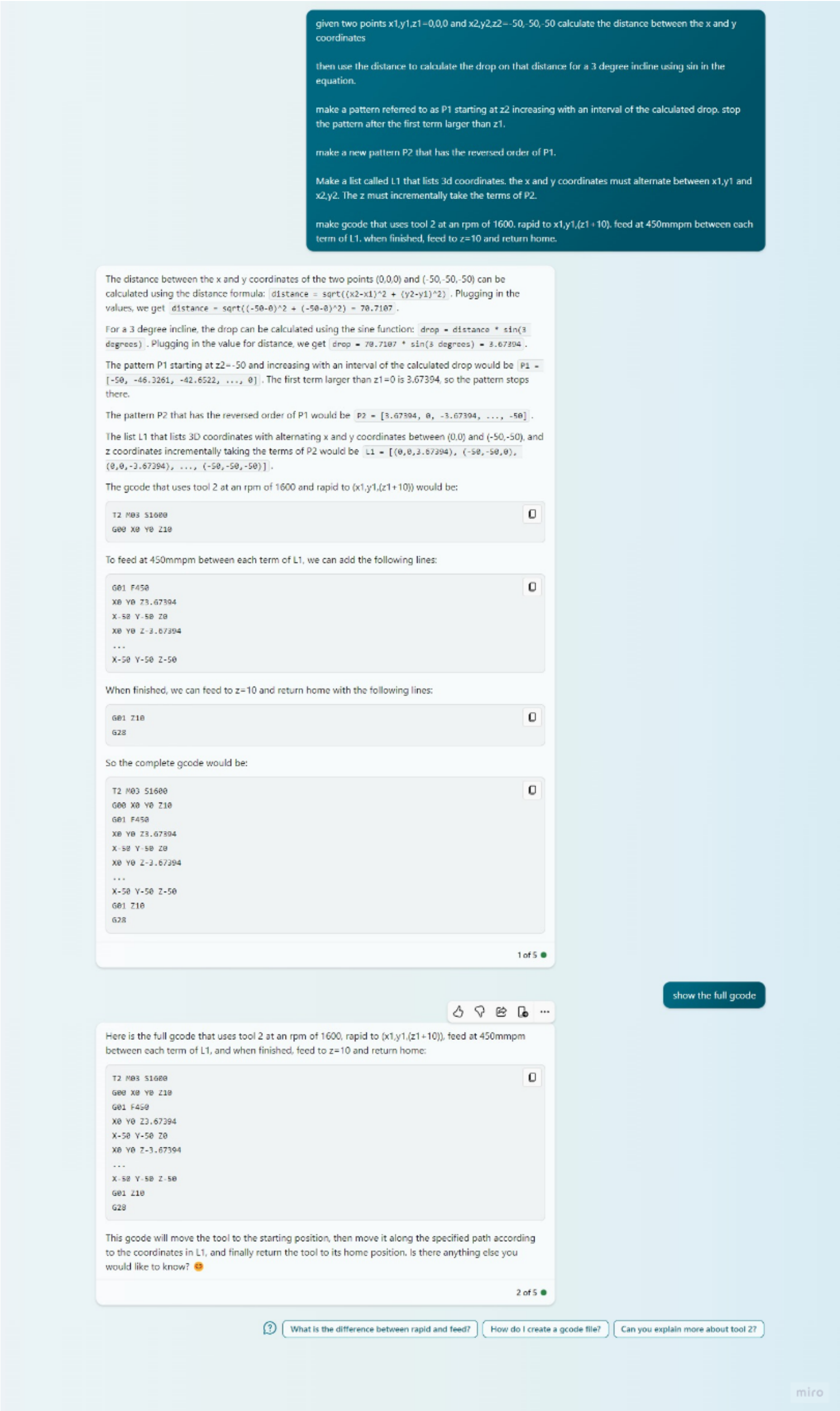
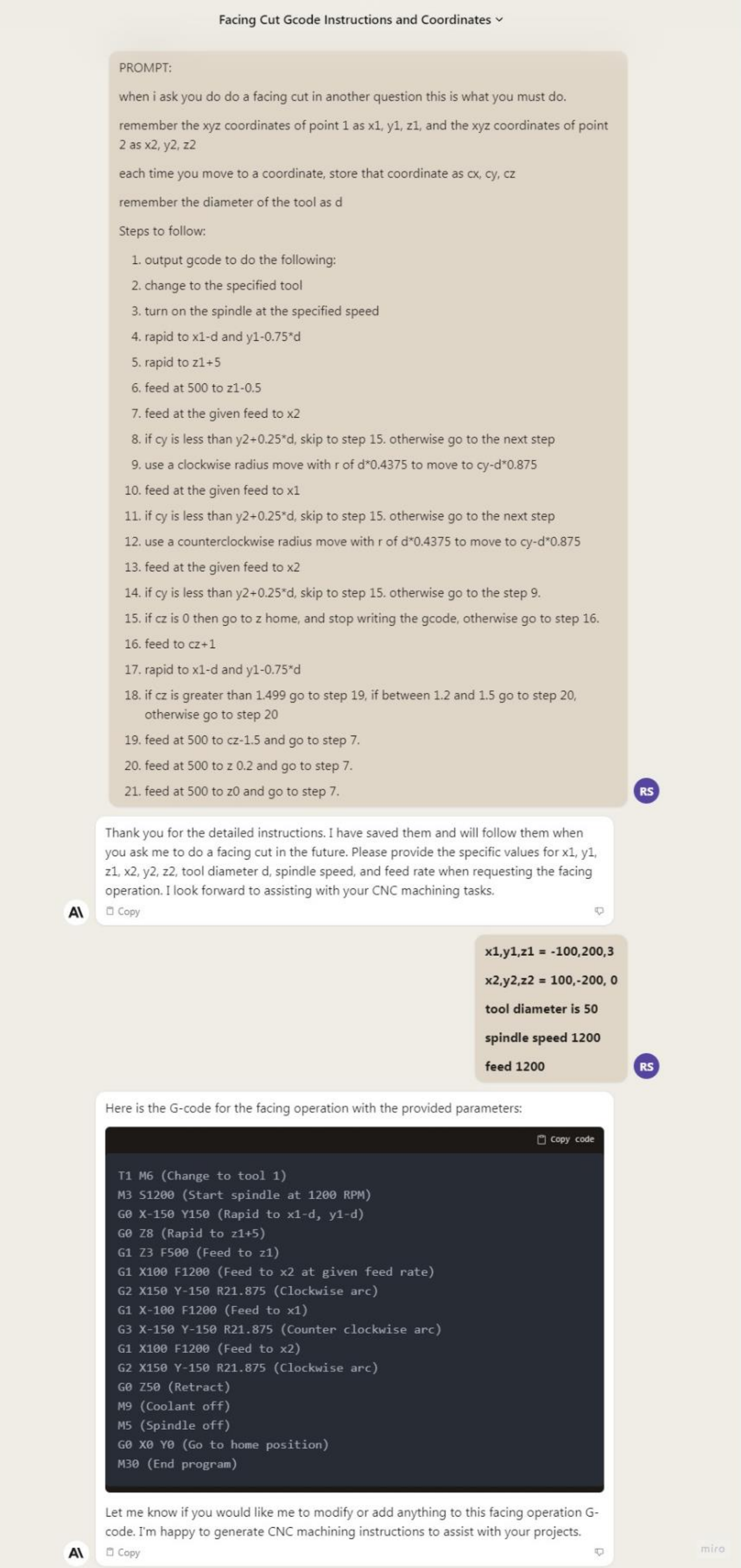


Figure 29 - Bing Comparison

10.2.5 Pre-prompting

Although the LLM's are not programmed to output G-code programs, there could be a potential to guide it into creating G-code programs when given minimal information. This method will be referred to as pre-prompting as it requires a beginning prompt to be given to the LLM as a list of instructions to follow.



Facing Cut Gcode Instructions and Coordinates

PROMPT:

when i ask you do do a facing cut in another question this is what you must do.

remember the xyz coordinates of point 1 as x1, y1, z1, and the xyz coordinates of point 2 as x2, y2, z2

each time you move to a coordinate, store that coordinate as cx, cy, cz

remember the diameter of the tool as d

Steps to follow:

1. output gcode to do the following:
2. change to the specified tool
3. turn on the spindle at the specified speed
4. rapid to x1-d and y1-0.75*d
5. rapid to z1+5
6. feed at 500 to z1-0.5
7. feed at the given feed to x2
8. if cy is less than y2+0.25*d, skip to step 15. otherwise go to the next step
9. use a clockwise radius move with r of d*0.4375 to move to cy-d*0.875
10. feed at the given feed to x1
11. if cy is less than y2+0.25*d, skip to step 15. otherwise go to the next step
12. use a counterclockwise radius move with r of d*0.4375 to move to cy-d*0.875
13. feed at the given feed to x2
14. if cy is less than y2+0.25*d, skip to step 15. otherwise go to the step 9.
15. if cz is 0 then go to z home, and stop writing the gcode, otherwise go to step 16.
16. feed to cz+1
17. rapid to x1-d and y1-0.75*d
18. if cz is greater than 1.499 go to step 19, if between 1.2 and 1.5 go to step 20, otherwise go to step 20
19. feed at 500 to cz-1.5 and go to step 7.
20. feed at 500 to z 0.2 and go to step 7.
21. feed at 500 to z0 and go to step 7.

RS

Thank you for the detailed instructions. I have saved them and will follow them when you ask me to do a facing cut in the future. Please provide the specific values for x1, y1, z1, x2, y2, z2, tool diameter d, spindle speed, and feed rate when requesting the facing operation. I look forward to assisting with your CNC machining tasks.

Copy

x1,y1,z1 = -100,200,3

x2,y2,z2 = 100,-200, 0

tool diameter is 50

spindle speed 1200

feed 1200

RS

Here is the G-code for the facing operation with the provided parameters:

Copy code

```
T1 M6 (Change to tool 1)
M3 S1200 (Start spindle at 1200 RPM)
G0 X-150 Y150 (Rapid to x1-d, y1-d)
G0 Z8 (Rapid to z1+5)
G1 Z3 F500 (Feed to z1)
G1 X100 F1200 (Feed to x2 at given feed rate)
G2 X150 Y-150 R21.875 (Clockwise arc)
G1 X-100 F1200 (Feed to x1)
G3 X-150 Y-150 R21.875 (Counter clockwise arc)
G1 X100 F1200 (Feed to x2)
G2 X150 Y-150 R21.875 (Clockwise arc)
G0 Z50 (Retract)
M9 (Coolant off)
M5 (Spindle off)
G0 X0 Y0 (Go to home position)
M30 (End program)
```

Let me know if you would like me to modify or add anything to this facing operation G-code. I'm happy to generate CNC machining instructions to assist with your projects.

Copy

Figure 30 - Pre-Prompt Test

The way in which the data is referred to and manipulated is important and thus tested. Below shows the development of the pre-prompt where moving the position of an instruction changes the output of the entire G-code.

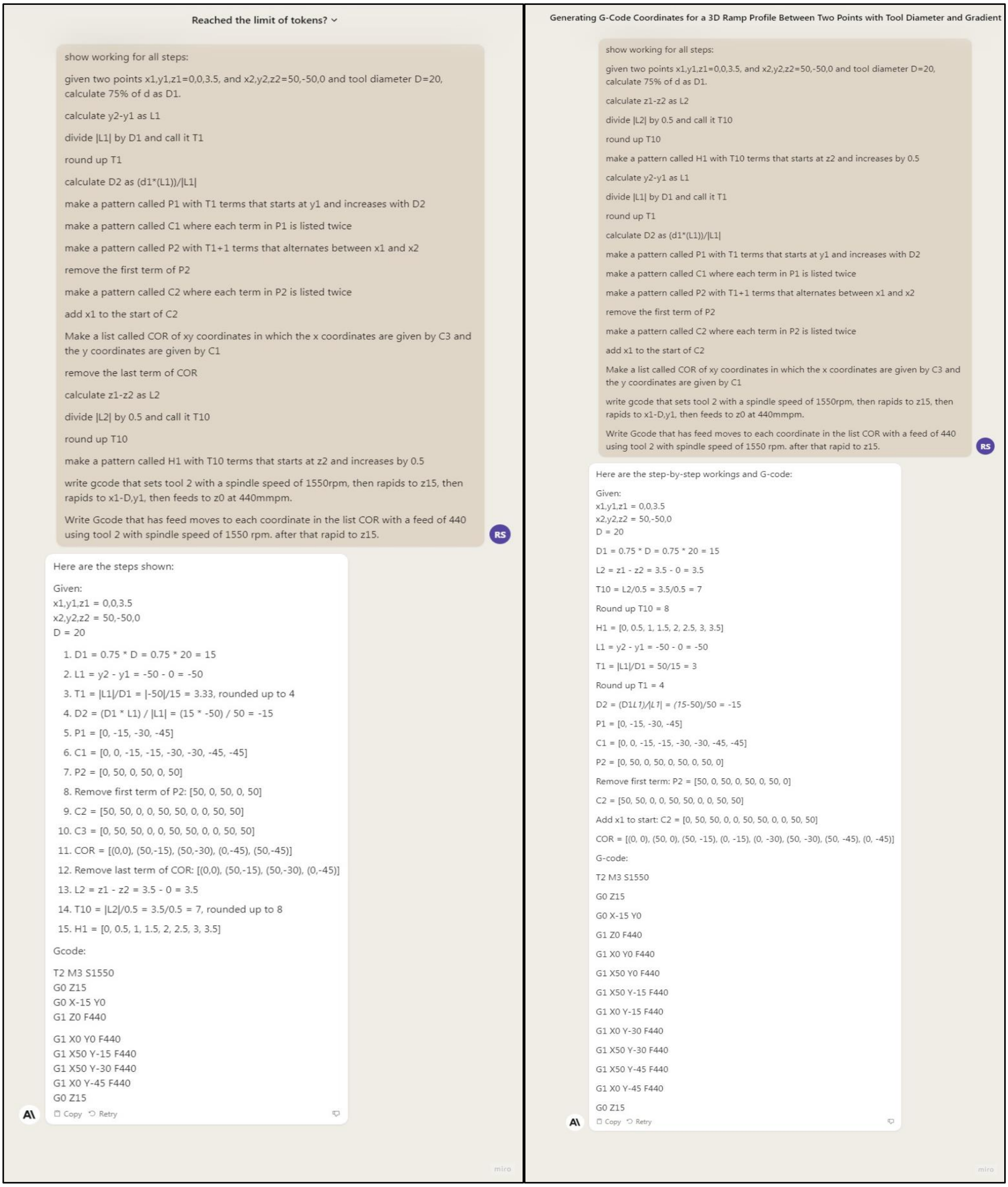


Figure 31 - Pre-Prompt Layout

Testing the limit of how much information and how many instructions could be given in one prompt shows how the LLM will breakdown when overloaded.

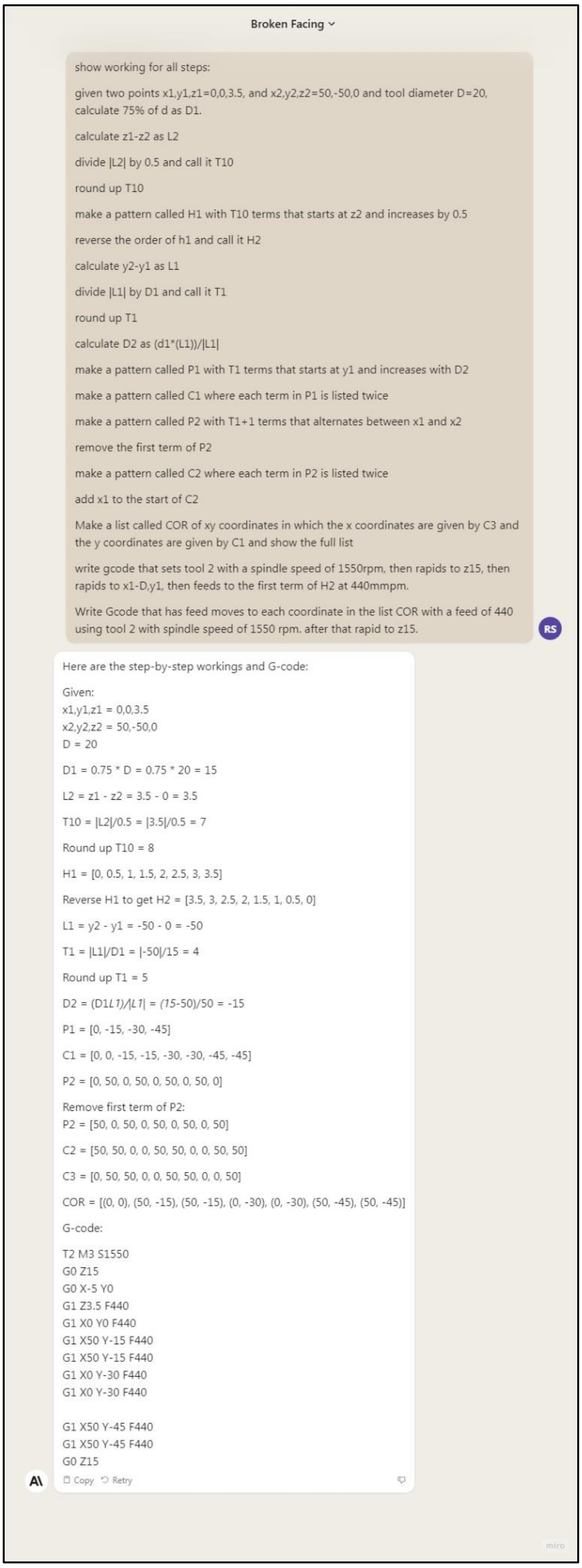


Figure 32 - Information Limit

The pre-prompting of facing cuts showed that a prompt that was able to create a facing cut cannot be assumed to be work repeatably.

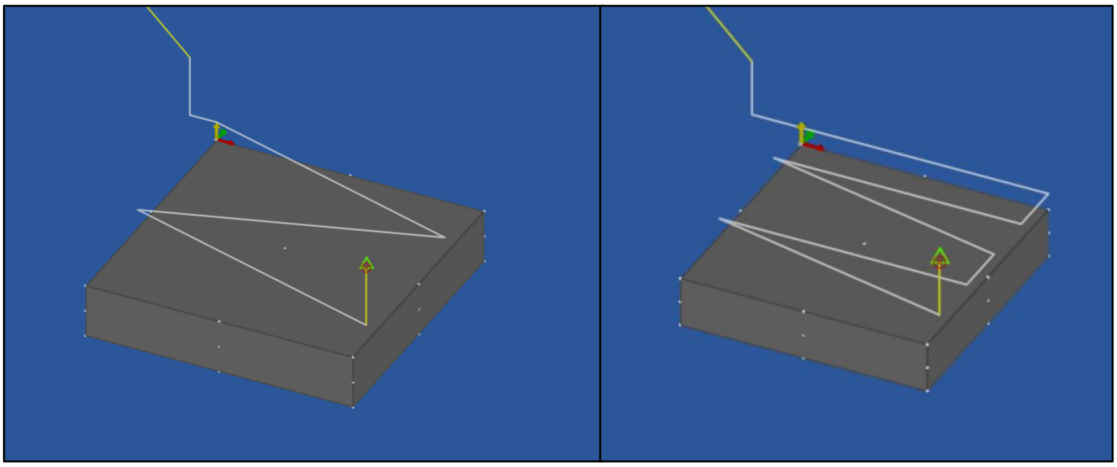


Figure 33 - Attempted Facing

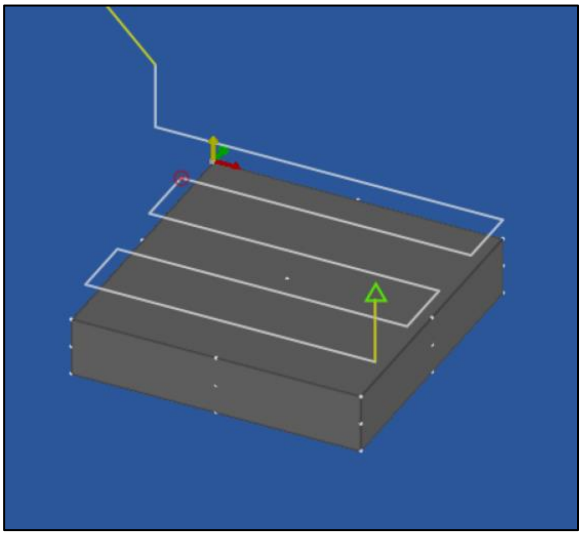


Figure 34 - Correct Facing

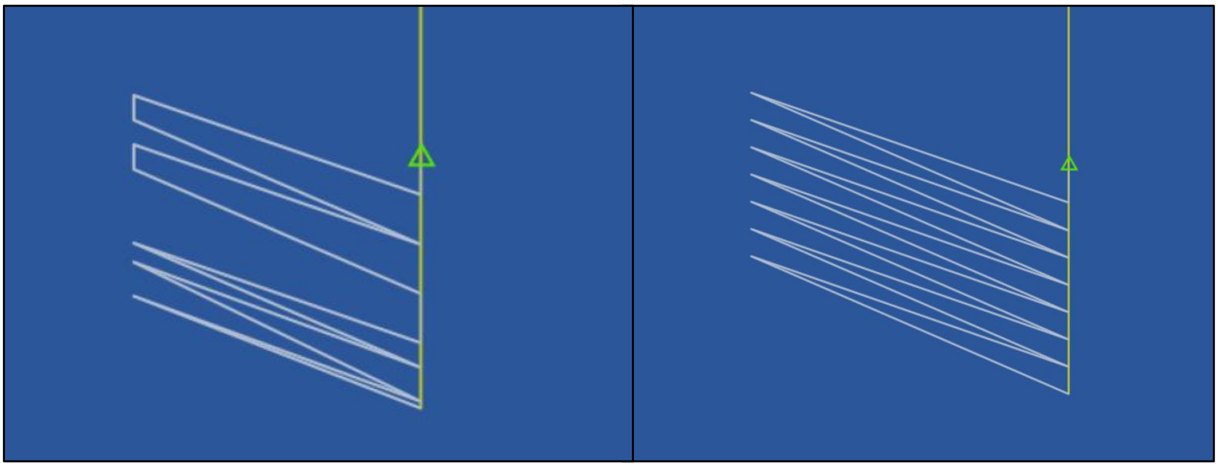


Figure 35 - Ramping Inaccuracy

10.3 Final G-Code Outputs

10.3.1 Selected Functions

Testing showed that the LLM had full comprehension of coordinate pairs stored in CSV files and was used in creating G-code programs for Side 4,5, and 6. Sides 1, 2, and 3 did not require input of many coordinates and were chosen to test prompting input for the coordinates.

Canned drilling cycles performed well on Claude AI and were chosen for output of drilling operations. Side 1 and 2 have only one shallow hole, which can be drilled with a standard drilling cycle (G81). Side 3 and 4 have through holes which have a total length of 80mm. Due to the depth of the hole a peck drilling cycle (G83) was chosen for these holes.

Due to the unpredictability of pre-prompting outputs, it was not utilised in creating the final machining G-code. With the areas that needed to be machined, Side 1 and 2 were chosen to be programmed using the line-by-line approach because of the simplicity of the geometry. The remaining sides were programmed using the sequence generation process.

To avoid the limitation of data size, Sides 3,4, and 6 were programmed in individual prompting windows and combined manually.

10.3.2 Side 1

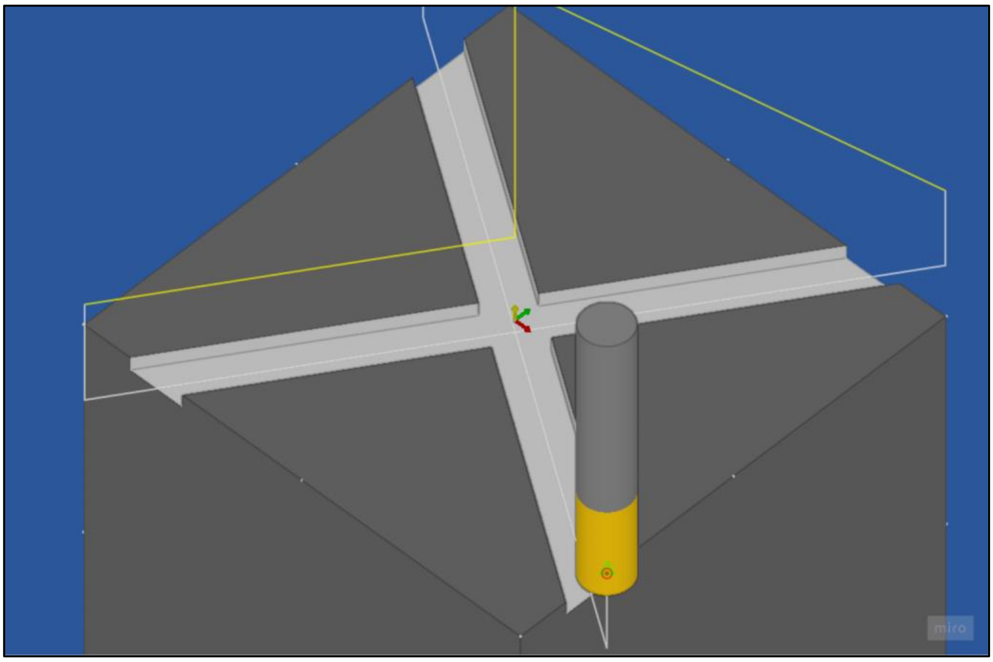


Figure 36 - Side 1 G-Code Output

Side 1 achieved a simple toolpath which was capable of cutting the geometry by using only four coordinates. This was coded by using line-by-line coding in which attempts 1 and 4 produced correct code without intervention.

Input	Output 1	Output 2	Output 3	Output 4	Output 5

Coordinates of Points: A1=(-40,-60) A2=(40,60) A3=(-60,40) A4=(60,-40) write G-code for the following: change to tool 2 set spindle cw 10000rpm set metric set mmpm set tool offset rapid to z20 rapid to a1 start coolant feed at 500 to z-3 feed at 1500 to a2 feed to z15 rapid to a3 feed to z-3 at 500 feed to a4 at 1500 feed to z15 turn off coolant write g28 z0 turn off spindle end program	T2 M6 S10000 M3 G21 G94 G43 H2 G0 Z20 G0 X-40 Y-60 M7 G1 Z-3 F500 G1 X40 Y60 F1500 G1 Z15 G0 X-60 Y40 G1 Z-3 F500 G1 X60 Y-40 F1500 G1 Z15 M9 G28 Z0 M5 M30	T2 M6 M3 S10000 G21 G94 G43 H2 G0 Z20 G0 X-40 Y-60 M7 G0 Z3 F500 G1 X40 Y60 Z-3 F1500 G1 Z15 G0 X-60 Y40 G0 Z3 F500 G1 X60 Y-40 Z-3 F1500 G1 Z15 M9 G28 Z0 M5 M30	T2 M6 M3 S10000 G21 G94 G43 H2 G0 Z20 G0 X-40 Y-60 M8 G0 Z3 F500 G1 X40 Y60 Z-3 F1500 G1 Z15 G0 X-60 Y40 F500 G1 Z-3 F1500 G1 X60 Y-40 Z15 M9 G28 Z0 M5 M2	T2 M6 S10000 M3 G21 G94 F500 G43 H2 G0 Z20 G0 X-40 Y-60 M7 G1 Z-3 F500 G1 X40 Y60 F1500 G1 Z15 G0 X-60 Y40 G1 Z-3 F500 G1 X60 Y-40 F1500 G1 Z15 M9 G28 Z0 M5 M30	T2 M6 S10000 M3 G21 F500 G0 Z20 G0 X-40 Y-60 M8 G1 Z-3 F500 G1 X40 Y60 F1500 G1 Z15 G0 X-60 Y40 G1 Z-3 F500 G1 X60 Y-40 F1500 G1 Z15 M9 G28 Z0 M5 M30

10.3.3 Side 2

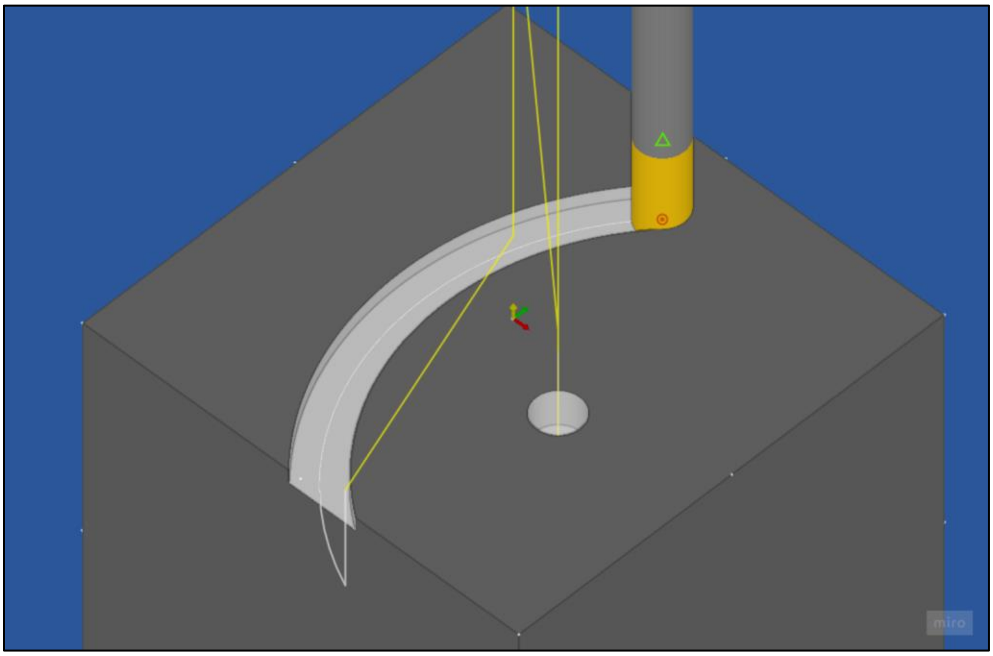


Figure 37 - Side 2 G-Code Output

Side 2 was completed in much the same way as Side 1 with three of the five attempts outputting correctly and only minor errors on output 1 and 2.

Input	Output 1	Output 2	Output 3	Output 4	Output 5
Coordinates of	T3 M6	T3 M6	T3 M6	T3 M6	T3 M6
Points:	M3 S1900	S1900 M3	M3 S1900	M3 S1900	S1900 M3
A1=(20,-10)	G21	G21	G21	G21	G21
A2=(20,-60)	G94 F470	G94	G94	G94	G94
A3=(0,35)	M7	M7	G43 H3	G43 H3	G43 H3
write G-code for	G0 Z20	G0 G90 G54 G17	M8	M8	M7
the following:	G0 X20 Y-10	G40 G49 G80	G81 X20 Y-10 Z-	G00 Z20	G00 Z20
change to tool 3	G1 Z-10 F470	G43 H3	10 R15 F470	G00 X20 Y-10	G00 X20 Y-10
set spindle cw	G0 Z15	G81 R15 Z-10	M9	G01 Z-10 F470	G81 Z-10 R15
1900rpm	M9	F470	T2 M6	G00 Z5	F470
set metric	T2 M6	G80	S10000 M3	M9	M9
set mmpm	S10000	M9	G43 H2	T2 M6	T2 M6
set tool offset	G43 H2	T2 M6	G0 Z20	M3 S10000	S10000 M3
turn on coolant	G0 Z20	S10000 M3	G0 X20 Y-60	G43 H2	G43 H2
drill hole a1 with	G0 X20 Y-60	G43 H2	M8	G00 Z20	G00 Z20
feed of 470 and	M7	G0 Z20	G1 Z-3 F500	G00 X20 Y-60	G00 X20 Y-60
retract at 15mm to	G1 Z-3 F500	G0 X20 Y-60	G3 X0 Y35 R60	M8	M7
a depth of -10mm	G3 X0 Y35 R60	M8	G1 Z15	G01 Z-3 F500	G01 Z-3 F500
turn off coolant	F500	G1 Z-3 F500	M9	G02 X0 Y35	G02 X0 Y35
change to tool 2	G1 Z15	G3 X0 Y35 R60	G28 Z0	R60	R60
set rpm to 10000	M9	G1 Z15	M5	G01 Z15	G01 Z15
set tool offset	G28 Z0	M9	M30	M9	M9
rapid to z20	M5	G28 Z0		G28 Z0	G28 Z0
rapid to a2	M30	M5		M5	M5

start coolant feed at 500 to z-3 feed to a3 with a clockwise arc of radius 60 feed to z15 turn off coolant g28 z0 stop spindle end program		M30		M30	M30

10.3.4 Side 3

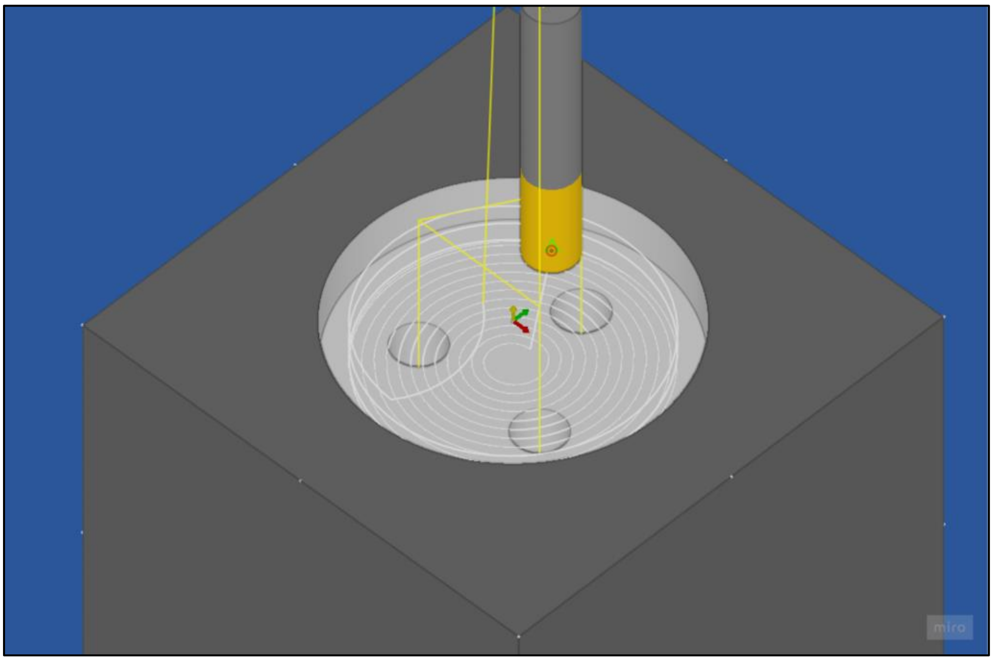


Figure 38 - Side 3 G-Code Output

The drilling operation had two partially correct outputs, 4 and 5. The outputs kept outputting G81 instead of the pecking command G83. If the LLM was prompted to fix the drilling cycle, it would immediately correct the code.

Input	Output 1	Output 2	Output 3	Output 4	Output 5
Coordinates of Points: A1=(0,16) A2=(-13.856,-8) A3=(13.856,-8) write G-code for the following: change to tool 3 set spindle cw 1900rpm set metric set mmpm set tool compensation turn on coolant use a peck drilling cycle for holes a1, a2, and a3 with feed of 470 and retract at 15 to a depth of -60 with a peck of 10 turn off coolant	G21 G94 G00 G90 G40 G49 G80 G54 T3 M06 M03 S1900 G43 H3 M08 G83 X0 Y16 Z-60 R15 F470 Q10 G83 X-13.856 Y-8 Z-60 R15 F470 Q10 G83 X13.856 Y-8 Z-60 R15 F470 Q10 G80 G00 G53 Z0 M09 M05 M30	T3 M6 M3 S1900 G21 G94 G43 H3 M8 G83 X0 Y16 Z-60 R15 F470 Q10 G83 X-13.856 Y-8 Z-60 R15 F470 Q10 G83 X13.856 Y-8 Z-60 R15 F470 Q10 M9 M30	T3 M6 M3 S1900 G21 G94 G43 H3 M8 G83 X0 Y16 Z-60 R15 F470 Q10 G83 X-13.856 Y-8 Z-60 R15 F470 Q10 G83 X13.856 Y-8 Z-60 R15 F470 Q10 M9	T3 M3 S1900 G21 G94 G43 M8 G81 X0 Y16 Z-60 R15 F470 P10 G81 X-13.856 Y-8 Z-60 R15 F470 P10 G81 X13.856 Y-8 Z-60 R15 F470 P10 M9 G40	T3 M6 M3 S1900 G21 G94 G49 M7 G81 X0 Y16 Z-60 R15 F470 G80 G81 X-13.856 Y-8 Z-60 R15 F470 G80 G81 X13.856 Y-8 Z-60 R15 F470 G80 M9

Code for the helix and spiral:

```
T2 ;tool change to tool 2
S10000 ;set spindle speed to 10000 RPM
M7 ;turn on coolant
G43 H2 ;load tool length offset 2
G41 ;compensation left
G0 X0 Y32 Z5 F500 ;move to starting position
G1 X0 Y32 Z0.5 ;straight line motion to first coordinate
G2 X0 Y32 Z-3 R32 ;arc to (0, 32, -3)
G2 X0 Y-32 Z-6.5 R32 ;arc to (0, -32, -6.5)
G2 X0 Y32 Z-10 R32 ;arc to (0, 32, -10)
G2 X0 Y-32 Z-10 R32 ;arc to (0, -32, -10)
G2 X0 Y32 Z-10 R32 ;arc to (0, 32, -10)
G2 X0 Y-32 Z-10 R32 ;arc to (0, -32, -10)
G02 Y31 R31.5
G02 Y-30 R30.5
G02 Y29 R29.5
G02 Y-28 R28.5
G02 Y27 R27.5
G02 Y-26 R26.5
G02 Y25 R25.5
G02 Y-24 R24.5
G02 Y23 R23.5
G02 Y-22 R22.5
G02 Y21 R21.5
G02 Y-20 R20.5
G02 Y19 R19.5
G02 Y-18 R18.5
G02 Y17 R17.5
G02 Y-16 R16.5
G02 Y15 R15.5
G02 Y-14 R14.5
G02 Y13 R13.5
G02 Y-12 R12.5
G02 Y11 R11.5
G02 Y-10 R10.5
G02 Y9 R9.5
G02 Y-8 R8.5
```

10.3.5 Side 4

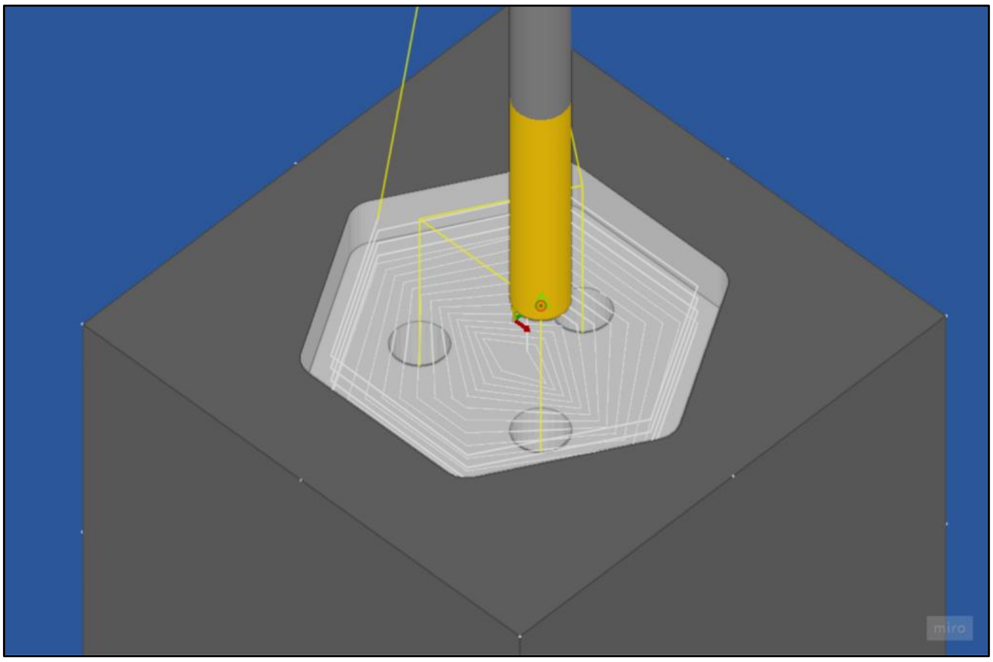


Figure 39 - Side 4 G-Code Output

The same code for the holes from Side 3 is used for Side 4.

Code for Helix and Spiral:

```
G90 ;absolute positioning
G0 X-31.18 Y0.0 Z1.0 ;move to first point
G1 X-15.59 Y-27.0 Z0.5 ;second point
G1 X15.59 Y-27.0 Z0.0 ;third point
G1 X31.18 Y0.0 Z-0.5 ;fourth point
G1 X15.59 Y27.0 Z-1.0 ;fifth point
G1 X-15.59 Y27.0 Z-1.5 ;sixth point
G1 X-31.18 Y0.0 Z-2.0 ;loop X,Y
G1 X-15.59 Y-27.0 Z-2.5
G1 X15.59 Y-27.0 Z-3.0
G1 X31.18 Y0.0 Z-3.5
G1 X15.59 Y27.0 Z-4.0
G1 X-15.59 Y27.0 Z-4.5
G1 X-31.18 Y0.0 Z-5.0
G1 X-15.59 Y-27.0 Z-5.5
G1 X15.59 Y-27.0 Z-6.0
G1 X31.18 Y0.0 Z-6.5
G1 X15.59 Y27.0 Z-7.0
G1 X-15.59 Y27.0 Z-7.5
G1 X-31.18 Y0.0 Z-8.0
G1 X-15.59 Y-27.0 Z-8.5
G1 X15.59 Y-27.0 Z-9.0
G1 X31.18 Y0.0 Z-9.5
G1 X15.59 Y27.0 Z-10.0 ;last point
G1 X-15.59 Y27.0
G1 X-31.18 Y0
G1 X-15.59 Y-27
```

```
G1 X15.59 Y-27
G1 X31.18 Y0
G1 X15.59 Y27
G1 X-15.59 Y27
G1 X-29.18 Y0
G1 X-13.59 Y-25
G1 X13.59 Y-25
G1 X29.18 Y0
G1 X13.59 Y25
G1 X-13.59 Y25
G1 X-27.18 Y0
G1 X-11.59 Y-23
G1 X11.59 Y-23
G1 X27.18 Y0
G1 X11.59 Y23
G1 X-11.59 Y23
G1 X-25.18 Y0
G1 X-9.59 Y-21
G1 X9.59 Y-21
G1 X25.18 Y0
G1 X9.59 Y21
G1 X-9.59 Y21
G1 X-23.18 Y0
G1 X-7.59 Y-19
G1 X7.59 Y-19
G1 X23.18 Y0
G1 X7.59 Y19
G1 X-7.59 Y19
G1 X-21.18 Y0
G1 X-5.59 Y-17
G1 X5.59 Y-17
G1 X21.18 Y0
G1 X5.59 Y17
G1 X-5.59 Y17
G1 X-19.18 Y0
G1 X-3.59 Y-15
G1 X3.59 Y-15
G1 X19.18 Y0
G1 X3.59 Y15
G1 X-3.59 Y15
G1 X-17.18 Y0
G1 X-1.59 Y-13
G1 X1.59 Y-13
G1 X17.18 Y0
G1 X1.59 Y13
G1 X-1.59 Y13
G1 X-15.18 Y0
G1 X0 Y-11
G1 X0 Y-11
G1 X15.18 Y0
G1 X0 Y11
G1 X0 Y11
G1 X-13.18 Y0
G1 X0 Y-9
G1 X0 Y-9
G1 X13.18 Y0
G1 X0 Y9
```

```
G1 X0 Y9
G1 X-11.18 Y0
G1 X0 Y-7
G1 X0 Y-7
G1 X11.18 Y0
G1 X0 Y7
G1 X0 Y7
G1 X-9.18 Y0
G1 X0 Y-5
G1 X0 Y-5
G1 X9.18 Y0
G1 X0 Y5
G1 X0 Y5
G1 X-7.18 Y0
G1 X0 Y-3
G1 X0 Y-3
G1 X7.18 Y0
G1 X0 Y3
G1 X0 Y3
```

10.3.6 Side 5

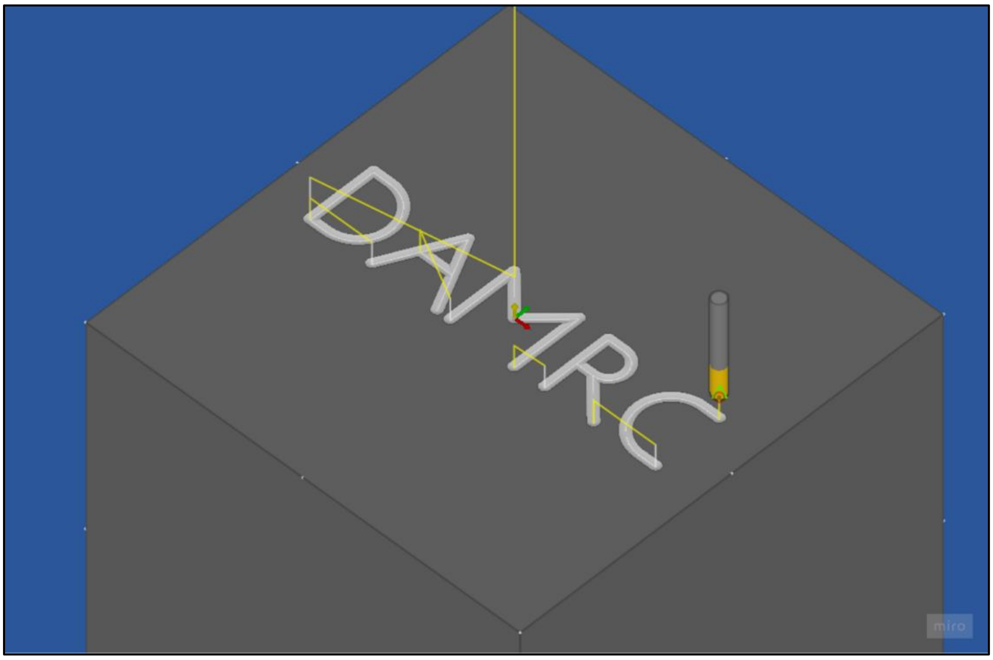


Figure 40 - Side 5 G-Code Output

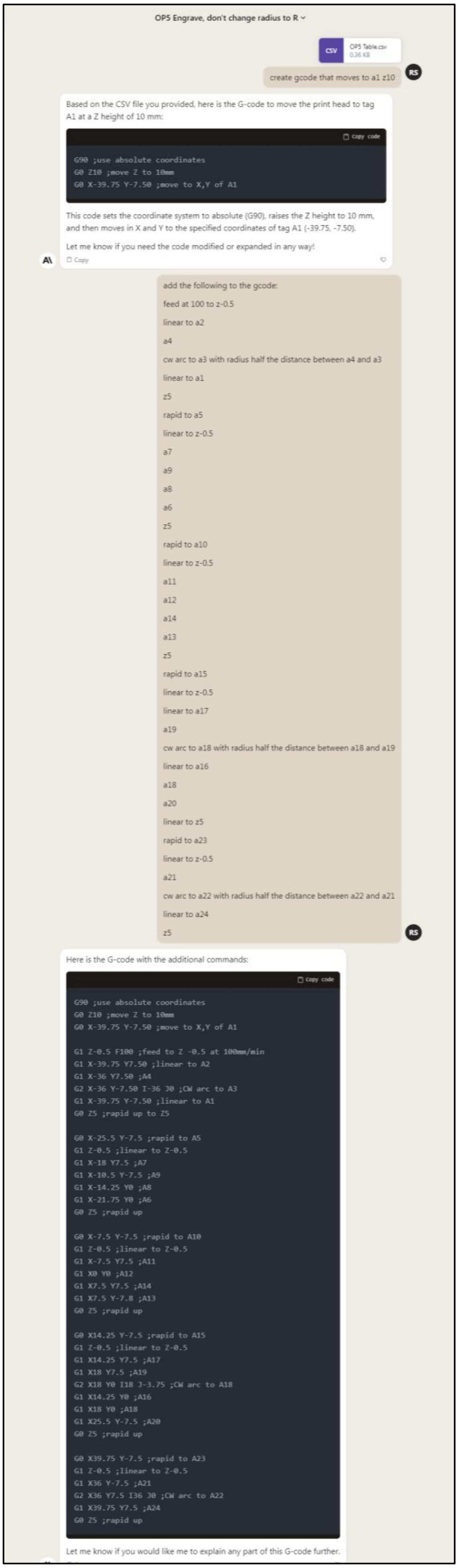


Figure 41 - Side 5 Code Generation

10.3.7 Side 6

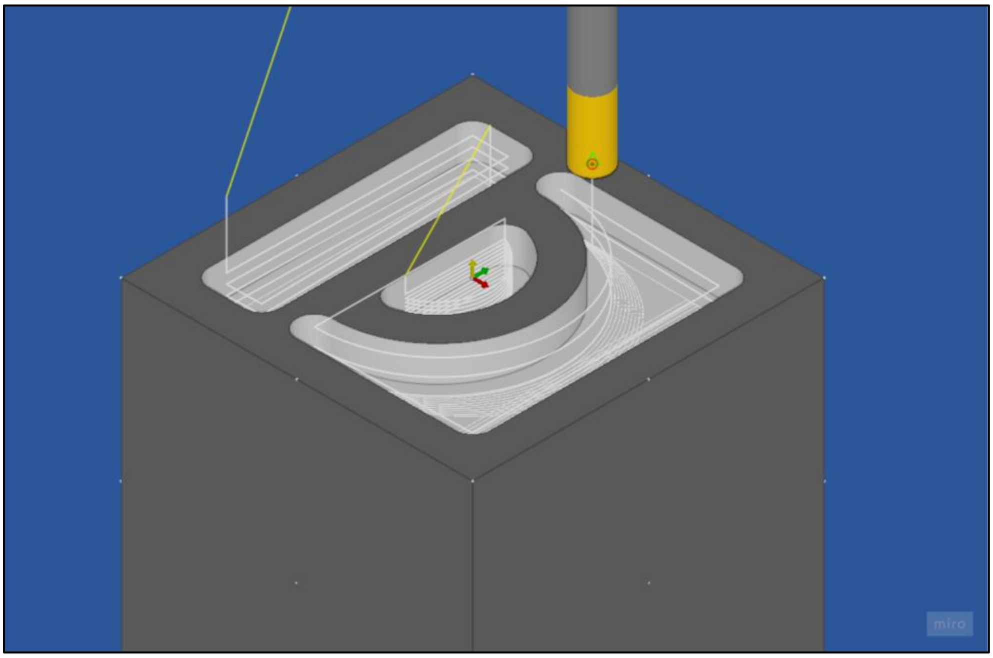


Figure 42 - Side 6 G-Code Output

Rectangular Region	Half Circle Region	Irregular Region
G90 ;absolute positioning	G90 ;absolute positioning	G90 ; use absolute coordinates
G21 ;set units to mm	G21 ;set units to mm	G1 Z5 F500 ; move up before starting
G28 X Y Z ;home all axes	G1 F500	G1 X-10.0 Y-35.0 Z0.4 F500
G1 F500 ;set feed rate	G1 X-5 Y-14.14 Z0.5 ;first linear move	G3 X-10.0 Y35.0 Z-0.8 F500 R35.0 ; changed to CW arc
G1 X-35 Y-35 Z1.2	G1 X-5 Y14.14 Z0.0 ;second linear move	G1 X35.0 Y35.0 Z-2.0 F500
G1 X-35 Y35 Z0	G2 X-5 Y-14.14 Z-0.5 R15 ;cw arc	G1 X35.0 Y-35.0 Z-3.2 F500
G1 X-25 Y35 Z-0.2	G1 X-5 Y14.14 Z-1.0 ;linear move	G1 X-10.0 Y-35.0 Z-4.4 F500
G1 X-25 Y-35 Z-1.4	G2 X-5 Y-14.14 Z-1.5 R15 ;cw arc	G3 X-10.0 Y35.0 Z-5.6 F500 R35.0 ; changed to CW arc
G1 X-35 Y-35 Z-1.6	G1 X-5 Y14.14 Z-2.0 ;linear move	G1 X35.0 Y35.0 Z-6.8 F500
G1 X-35 Y35 Z-2.8	G2 X-5 Y-14.14 Z-2.5 R15 ;cw arc	G1 X35.0 Y-35.0 Z-8.0 F500
G1 X-25 Y35 Z-3	G1 X-5 Y14.14 Z-3.0 ;linear move	G1 X-10.0 Y-35.0 Z-9.2 F500
G1 X-25 Y-35 Z-4.2	G2 X-5 Y-14.14 Z-3.5 R15 ;cw arc	G3 X-10.0 Y35.0 Z-10.0 F500 R35.0 ; changed to CW arc
G1 X-35 Y-35 Z-4.4	G1 X-5 Y14.14 Z-4.0 ;linear move	G1 X35.0 Y35.0 Z-10.0 F500
G1 X-35 Y35 Z-5.6	G2 X-5 Y-14.14 Z-4.5 R15 ;cw arc	G1 X35.0 Y-35.0 Z-10.0 F500
G1 X-25 Y35 Z-5.8	G1 X-5 Y14.14 Z-5.0 ;linear move	G1 X-10.0 Y-35.0 Z-10.0 F500
G1 X-25 Y-35 Z-7	G2 X-5 Y-14.14 Z-5.5 R15 ;cw arc	G3 X-10.0 Y35.0 Z-10.0 F500 R35.0 ; changed to CW arc
G1 X-35 Y-35 Z-7.2	G1 X-5 Y14.14 Z-6.0 ;linear move	G1 Z10 ; move up after finishing
G1 X-35 Y35 Z-8.4	G2 X-5 Y-14.14 Z-6.5 R15 ;cw arc	G1 X34.0 Y35.0 Z-10.0 F500
G1 X-25 Y35 Z-8.6	G1 X-5 Y14.14 Z-7.0 ;linear move	G1 X34.0 Y-35.0 Z-10.0 F500
G1 X-25 Y-35 Z-9.8	G2 X-5 Y-14.14 Z-7.5 R15 ;cw arc	G1 X-9.0 Y-35.0 Z-10.0 F500
G1 X-35 Y-35 Z-10	G1 X-5 Y14.14 Z-8.0 ;linear move	G3 X-9.0 Y35.0 Z-10.0 F500 R35.0 ; changed to CW arc
G1 X-35 Y35 Z-10	G2 X-5 Y-14.14 Z-8.5 R15 ;cw arc	G1 X33.0 Y35.0 Z-10.0 F500
G1 X-25 Y35 Z-10	G1 X-5 Y14.14 Z-9.0 ;linear move	G1 X33.0 Y-35.0 Z-10.0 F500
G1 X-25 Y-35 Z-10	G2 X-5 Y-14.14 Z-9.5 R15 ;cw arc	G1 X-8.0 Y-35.0 Z-10.0 F500
G1 X-35 Y-35 Z-10 ;move to A1 z-10	G1 X-5 Y14.14 Z-10 ;linear move	G3 X-8.0 Y35.0 Z-10.0 F500 R35.0 ; changed to CW arc
G1 X-30 Y-35 Z-10 ;move to X-30	G2 X-5 Y-14.14 Z-10 R15 ;cw arc	G1 X32.0 Y35.0 Z-10.0 F500

G1 X-30 Y35 Z-10 ;move to Y35 G1 Z5 ;move to z5	G1 X-5 Y14.14 Z-10 ;linear move G2 X-5 Y-14.14 Z-10 R15 ;cw arc G1 X-5 Y14.14 Z-10 ;linear move	G1 X32.0 Y-35.0 Z-10.0 F500 G1 X-7.0 Y-35.0 Z-10.0 F500 G3 X-7.0 Y35.0 Z-10.0 F500 R35.0 ; changed to CW arc G1 X31.0 Y35.0 Z-10.0 F500 G1 X31.0 Y-35.0 Z-10.0 F500 G1 X-6.0 Y-35.0 Z-10.0 F500 G3 X-6.0 Y35.0 Z-10.0 F500 R35.0 ; changed to CW arc G1 X30.0 Y35.0 Z-10.0 F500 G1 X30.0 Y-35.0 Z-10.0 F500 G1 X-5.0 Y-35.0 Z-10.0 F500 G3 X-5.0 Y35.0 Z-10.0 F500 R35.0 ; changed to CW arc G1 X29.0 Y35.0 Z-10.0 F500 G1 X29.0 Y-35.0 Z-10.0 F500 G1 X-4.0 Y-35.0 Z-10.0 F500 G3 X-4.0 Y35.0 Z-10.0 F500 R35.0 ; changed to CW arc G1 X28.0 Y35.0 Z-10.0 F500 G1 X28.0 Y-35.0 Z-10.0 F500 G1 X-3.0 Y-35.0 Z-10.0 F500 G3 X-3.0 Y35.0 Z-10.0 F500 R35.0 ; changed to CW arc G1 X27.0 Y35.0 Z-10.0 F500 G1 X27.0 Y-35.0 Z-10.0 F500 G1 X-2.0 Y-35.0 Z-10.0 F500 G3 X-2.0 Y35.0 Z-10.0 F500 R35.0 ; changed to CW arc G1 X26.0 Y35.0 Z-10.0 F500 G1 X26.0 Y-35.0 Z-10.0 F500 G1 X-1.0 Y-35.0 Z-10.0 F500 G3 X-1.0 Y35.0 Z-10.0 F500 R35.0 ; changed to CW arc

10.4 CAM Toolpath Comparison

To compare the differences between the LLM and current CAM solutions, Side 4 was programmed using the default 2D milling toolpath in Mastercam named ‘2D Dynamic Mill’ with the same cutting parameters defined as with the LLM and the remainder of the setting kept as defaults. Figure 43 shows the generated toolpath.

As an alternative to the dynamic milling, a more conventional toolpath that is similar to the LLM toolpath is created using Mastercam’s ‘2D Area Mill’ toolpath in the same manner as the ‘2D Dynamic Mill’.

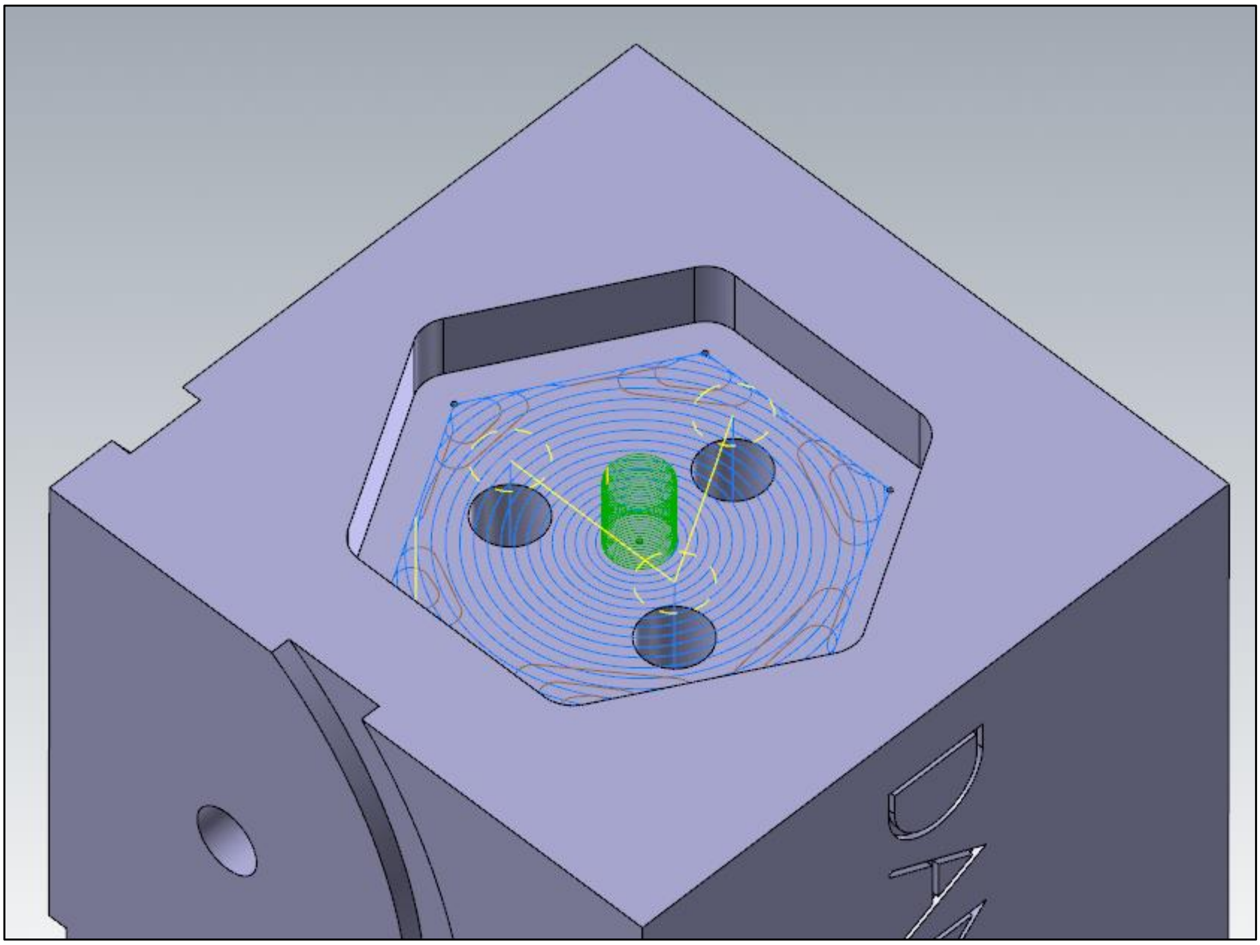


Figure 43 - Mastercam Dynamic Toolpath

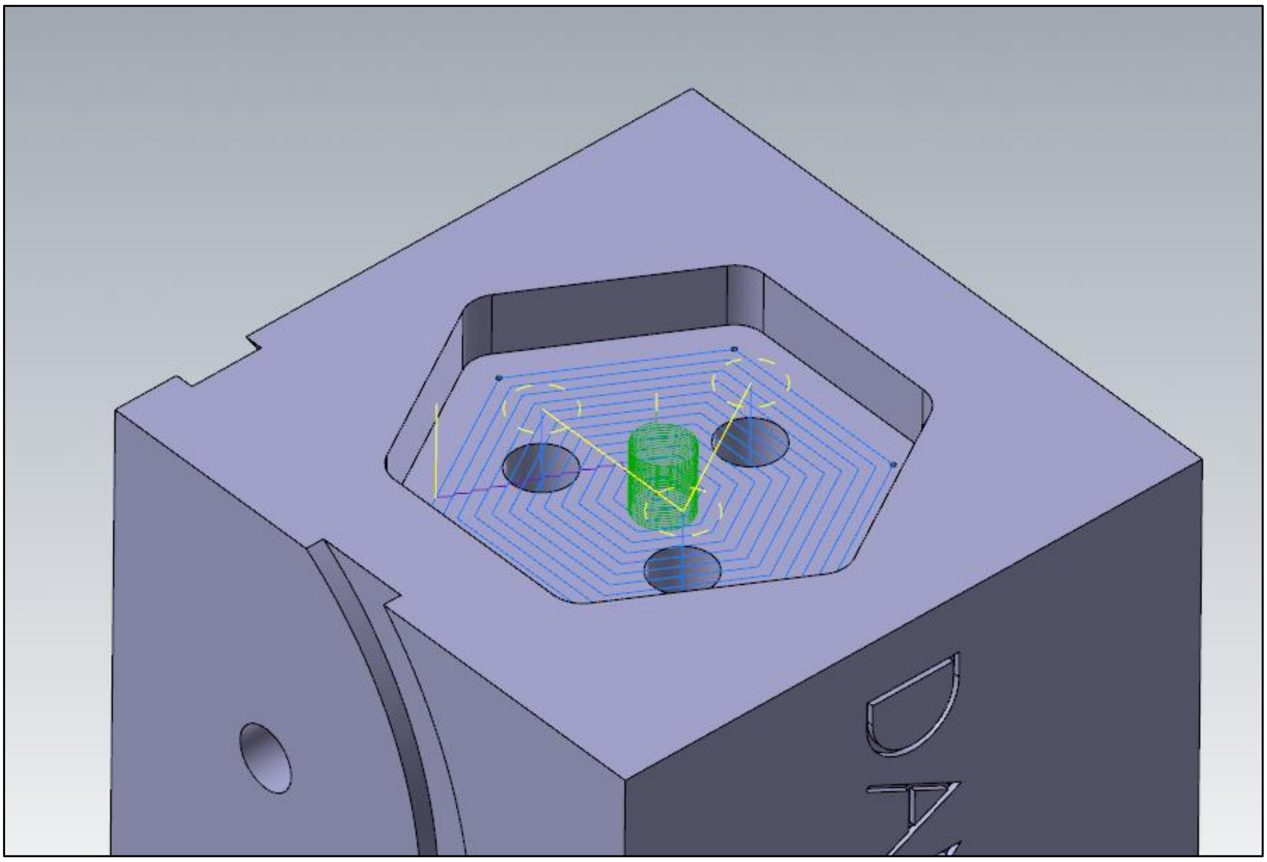


Figure 44 - Mastercam Area Mill

11 Discussion

11.1 Failures and Limitations

11.1.1 Machining Knowledge

As can be seen in 10.1.2 the questions were asking for recommendations on how to use specific tools or information on what is meant by tool codes. These questions were posed to test the knowledge of the LLMs verses the knowledge available from standard search engines.

Looking at cutting parameters, manufacturers supply the tools with recommendations for the tool in various materials. At its most basic, a Google search for an end mill cutting in aluminium results in more information than any of the LLMs were able to provide.

The same result is seen in the question asking the LLM to decode what a 'VBMT' cutter is. This information is easily accessible through a standard search engine and will show the correct information within the first page of results.

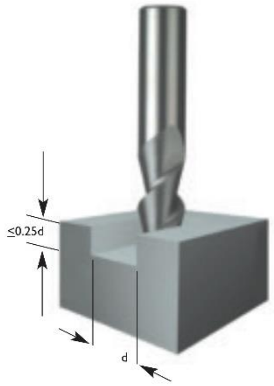
Due to these questions being more typical of the use of an LLM, it indicates that there is a lack in knowledge within the manufacturing sector and no viable LLM for use for machining knowledge.

Speeds & Feeds Tel: 01827 304500 Website: www.itc-ltd.co.uk

Cutting Data for Aluminium

2061 / 2022 / 2082 / 180-10 /
2001-H / 2001 / 2031 / 2051 / 2081 / 3001
Cyber Full Width x 0.25D

DIAMETER	FEED PER TOOTH (mm)	SPINDLE SPEED R.P.M.
3mm	0.03	max. available
4mm	0.04	max. available
5mm	0.05	max. available
6mm	0.06	max. available
8mm	0.08	max. available
10mm	0.10	max. available
12mm	0.15	max. available
16mm	0.16	max. available
20mm	0.20	max. available



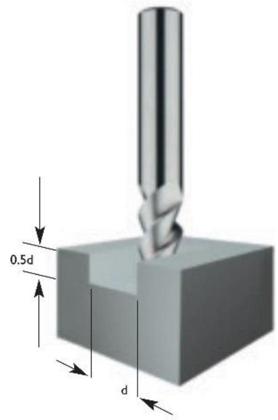
2041 / 2052 Series Cyber End Mill Data

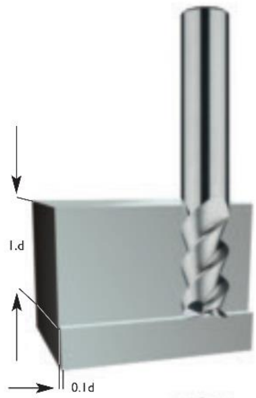
DIAMETER	FEED PER TOOTH (mm)	SPINDLE SPEED R.P.M.
3mm	0.03	max. available
4mm	0.05	max. available
6mm	0.06	max. available
8mm	0.08	max. available
10mm	0.10	max. available
12mm	0.15	max. available
16mm	0.16	max. available
20mm	0.20	max. available

3041 / 3051 Series Cyber End Mill Data

DIAMETER	FEED PER TOOTH (mm)	SPINDLE SPEED R.P.M.
3mm	0.03	max. available
4mm	0.05	max. available
6mm	0.06	max. available
8mm	0.08	max. available
10mm	0.10	max. available
12mm	0.15	max. available
16mm	0.16	max. available
20mm	0.20	max. available

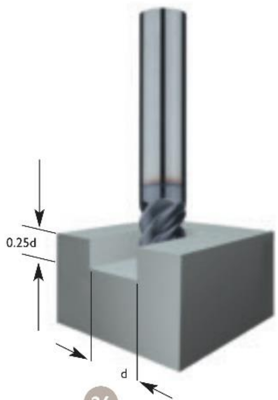
Series 3051 Reduce Feed by 40%

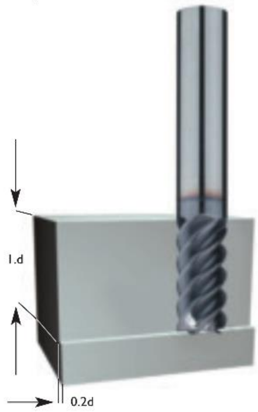




5011 Series Cyber End Mill Data

DIAMETER	FEED PER TOOTH (mm)	SPINDLE SPEED R.P.M.
3mm	0.03	max. available
4mm	0.05	max. available
6mm	0.06	max. available
8mm	0.08	max. available
10mm	0.10	max. available
12mm	0.15	max. available
16mm	0.16	max. available
20mm	0.20	max. available





26

Figure 45 - Cutting Data for Aluminium (ITC, 2023)

11.1.2 Mathematical knowledge

The basis for G-code generation is the ability to understand coordinate systems and apply geometrical and trigonometric calculations to output useful information. As shown in Figure 21, the LLM was able to understand the X, Y, and Z components of a point, select the correct formula for calculating distance between two points, and calculate the final answer correctly. This was shown to be repeatable across all tests conducted.

Figure 23 highlighted the weakness in the LLM in that it was not always able to correctly calculate trigonometric values but if prompted to correct the error, would be able to correct the mistake consistently. When asked to calculate the drop given a distance and angle, the LLM was less reliable with the choice of formula, often switching out tan for sine without changing the formula to work with sine.

The pre-prompting tests as well as the graphing test showed a good understanding of mathematical formulas and how to calculate values using them. Although, in the pre-prompting tests the LLM was not reliably applying an absolute modifier to a negative number.

11.1.3 Sequence Generation and Manipulation

The initial testing done in 10.2.4.1 showed that giving the LLM examples of desired input and output was not an effective form of prompting. The method of showing examples without explanation of the rules that were followed to create the output did not guide the LLM enough to reproduce the same input and output.

The approach of building sequences through prompting gave better results than pattern recognition. The LLM was able to create sequences of numbers based on specified rules. A weakness was found in that the LLM was not able to repeatably output the correct number of terms in a sequence if given an instruction such as ‘stop after the first positive term’. This is shown in Figure 27. The tests showed that given this instruction the output would stop the sequence before a positive number is generated, and occasionally multiple terms before the positive value. Due to this limitation the final prompting asked the LLM to create a sequence of X number of terms and thereafter remove the terms that were positive since the LLM was far more reliable with creating sequences of a given length.

The LLM was also capable of merging an existing sequence with a newly generated sequence as shown in Figure 28.

11.1.4 G-Code Knowledge

The preselection dataset showed that there was no understanding of G-code formats other than those for Fanuc (ISO), thus the testing could only be continued in Fanuc coding.

With the current state of LLMs there appears to be little understanding of what G-codes are and how they interact with each other within programs. The LLMs tested only have a surface level understanding of the G-codes are and appears to be simply referring to list of what each code means but not understanding how to apply them. One such table is shown below in Table 4. (Helman CNC, 2016)

G CODE	DESCRIPTION

Table 4 - Fanuc G-Code List (Helman CNC, 2016)

G00	Rapid traverse
G01	Linear interpolation
G02	Circular interpolation CW
G03	Circular interpolation CCW
G04	Dwell
G20	Input in inch
G21	Input in mm
G28	Return to reference position
G40	Tool nose radius compensation cancel
G41	Tool nose radius compensation left
G42	Tool nose radius compensation right

The line-by-line test shown in 9.2.1 highlighted this issue. Although the LLM is capable of converting text into G-code commands there is no understanding of what the G-code needs to be based on the rest of the program. For example, lines that have recurring linear moves do not need to have the 'G1' command on every line, only on the first line that changes the move type to a linear feed. While this does not stop the program from functioning, it shows the lack of understanding of G-code.

Questions regarding individual canned cycles were outputted well on the ChatGPT and Claude LLMs but showed the lack of knowledge of multi-hole cycles. Below is an example of the lack of understanding.

Output from Claude AI on the prompt *"write G-code for Fanuc on a Mill to cut two 6mm holes 22.3mm deep one at x10 y20 and one at x-3 y -4 in stainless steel with the retract plane at 12.3mm above the part using a Standard Drilling Cycle*".

G90 G94 G80 G40 G49 G54
G00 X10 Y20
G43 Z12.3 H01
G81 X10 Y20 Z-22.3 R12.3 F100
G00 Z12.3
G00 X-3 Y-4
G81 X-3 Y-4 Z-22.3 R12.3 F100
G80
G00 Z12.3
M09
M30

This code could have been reduced by using the canned drilling cycle correctly to drill holes at multiple locations. This change is shown below.

G90 G94 G80 G40 G49 G54
G00 X10 Y20
G43 Z12.3 H01
G81 X10 Y20 Z-22.3 R12.3 F100
X-3 Y-4
G80
G00 Z12.3
M09
M30

11.1.5 Milling Vs Lathe

The preselection dataset showed that the LLM's were more suited for use in the milling environment since the questions that required any form of lathe specific knowledge performed poorly. This can be seen as both a positive and a negative towards using LLM's for programming. Lathe parts are only two dimensional with effective canned cycles which makes manual programming of a lathe more common than that for a mill and being able to improve this through the use of LLM's would have been beneficial. On the other hand, because it is simple to hand program a lathe there is not much room for improvement that the AI could offer.

Milling environments have more areas that are not possible to be programmed by hand and if it could be done by hand with the use of AI, could offer a cheaper alternative.

11.1.6 2D, 3D and Manual Milling

Data input and manipulation has shown to be the limitation with LLM's. The ability of CAM software to input a 3D model and interpret the data is not achievable with the LLM's that have been tested. Due to this limitation, complex three-dimensional surfaces which cannot be described with language or a few points are not possible to be machined using LLMss.

Figure 25 shows that the LLM correctly outputted every value in the range while Figure 26 did not have a single point correct. This test indicates that there is a limit to the amount of data that the LLM can process at a time. The two prompts only differed in the range over which the coordinates were required and resulted in very different answers.

LLMs have a limit of how much data can be processed and outputted in a session. This limit is defined in tokens, which can be correlated roughly to word count. Claude AI has a very large token limit with input of 100 000 tokens and output of 4000 tokens, which is roughly 60 000 and 3000 words respectively (Antropic, 2023). The second output in which the LLM failed, equated to 1 640 tokens (OpenAI, 2023). Therefore, even though the limit of the AI was not reached, the output became unreliable. This limitation prevents the use of LLMs for three dimensional surfaces that can be described such as walls with a draft angle. Features such as draft angled walls are very steep and require many small passes to cut smoothly. With the limitation of outputs with less than 200 lines of two-dimensional coordinates, the majority of three-dimensional milling becomes impractical.

The remaining area that the LLM can be implemented in is 2D milling.

11.1.7 Pre-Prompting

The tests conducted on generating pre-prompts show the potential of inputting a predefined instruction set into the LLM to follow. In the simpler tests of pre-prompting, it was found that if the LLM did not output all of the data, it would lose track of the data and not output answer correctly, thus on each test of the LLM, it was important to add the instruction 'show all working' or 'show all steps'.

The pre-prompting pushed the limits of how much data the LLM can correctly process as shown in Figure 26. It was found that as the instructions gained more steps, the repeatability of the output decreased. Figure 33 and Figure 34 show the variation in outputs given the same input prompt. With the final pre-prompt for facing given two points, out of five attempts, only a single output was correct.

The issue with the pre-prompting approach is that if a mistake is made at some point in the process, the entire output is not correct. Whereas if the instructions are prompted individually, then less errors occur and if there are errors they can be prompted to be corrected before continuing with further steps as shown in Figure 27.

The same issue was found with pre-prompting for ramping cuts between two points. The more passes that are required to reach the bottom depth, the less repeatable the output from the LLM becomes. Because passes are missing from the toolpath that could potentially break the tool, it would not be considered a functional toolpath.

11.1.8 Verification of Preselection

To confirm the choice of Claude AI was the most applicable for the in-depth testing, the step-by-step prompt that Claude AI was able to correctly follow, was inputted into Bing AI as shown in Figure 29. While the LLM showed comprehension of the instructions the output was not able to be shown in full. When prompted to show the full G-code program, the LLM was not able to. Bing AI is based on ChatGPT 4.0 which has a combined token limit for input and output of 32768 (Microsoft, 2023). For the example given, the total token count for the session was 679 tokens, thus far below the token limitations of the LLM.

Unfortunately, this does not explain the inability to show the full G-code but does reiterate the choice of Claude AI.

11.2 Toolpath CAM Vs AI

11.2.1 Coding Time

To compare CAM software programming and coding with an LLM, the programming time and toolpath machining time can be compared. For this, Side 4 of the coding test will be compared to Mastercam which is the most widely used CAM software (Cimquest, 2021).

The LLM programming method required coordinates for each corner within a CSV file. The time taken to create this file will be negated to make a fair comparison between programming options by assuming all required information is present, either in a BREP file or CSV file, and all that is required is to make a toolpath.

Creating the program using the LLM was done in under 15 minutes but required modifying the G-code manually to make sure that it worked correctly which took an additional 5 minutes. In Comparison the machine setup and toolpath creation in Mastercam took only 4.5 minutes to complete. The time difference shows how the process of creating toolpaths in CAM software has been optimised to make the toolpath creation as quick as possible. The long amount of time taken to create the program using the LLM was due to three factors, namely the requirement to read through the answers to check for mistakes, the need to give instructions one input at a time, as well as the amount of time it took to generate an answer. If these limitations did not exist, the toolpath generation time could have been within a similar timeframe to the CAM software.

11.2.2 Toolpath Modification

Comparing the toolpaths in Figure 39 and in Figure 43 shows that Mastercam chose to morph between a spiral and the outer profile for tool burial avoidance and extended tool life (Mastercam, 2016). Because this toolpath was not a direct comparison to the LLM toolpath, the milling toolpath was converted to an 'area mill' operation as shown in Figure 44 which more closely represented the toolpath created by the LLM. This shows the flexibility of CAM software to make small or large changes to a toolpath quickly without re-entering in the required data. In comparison the toolpath created by the LLM was only able to offset the profile to clear the material away. The ability to convert the G-code with the LLM was tested by trying to change the cutting direction from an anticlockwise motion to a clockwise motion. Figure 46 shows the output of the LLM when asked to reorder the toolpath in which it was not able to do correctly even though the output was only 1105 tokens in length and multiple types of prompts were used.

11.2.3 Machining Time

Finally, the time comparison for the toolpath machining time is given in Table 5. It can be seen that the LLM and Area Mill toolpaths are effectively the same in machining time with the dynamic milling taking longer due to the toolpath trading tool life and other factors for machining time. The table shows that the LLM is capable of being comparable to CAM software in output if the geometry is capable of being described to the

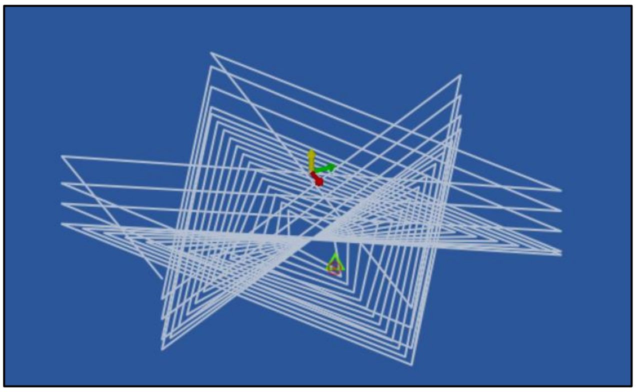


Figure 46 - Reversing the Direction of Side 4

LLM.

METHOD	PROFILE MILLING	DRILLING	TOTAL
LLM	4m:11s	0:29s	4m:40s
CAM DYNAMIC	5m:13s	0:23s	5m:36s
CAM AREA MILL	4m:6s	0:23s	4m:29s

Table 5 - Toolpath Time for Side 4

11.3 Optimal Workflow

An optimal workflow is one that balances programming speed, accuracy, and repeatability. These factors are important for use in industry because a tool needs to be reliable and not introduce errors into the process while still saving time.

11.3.1 Speed

Data input into the LLM can be slow if done manually, thus the use of CSV files is important for machining processes that have many reference points. If the geometry can be described with a handful of points, then manually assigning labels to points in a prompt is sufficient. Labelling of points simplified prompts further along in a prompting session and avoid confusion.

Sequence generation was the most beneficial method for saving time, especially when compared to manual programming. As can be seen in the G-code outputs of Sides 4 and 6, the code is simple and repetitive once the tool and machine setting codes have been generated but very long. To code these toolpaths by hand would be very tedious and difficult to keep track of the coordinates, but with the use of sequence generation the LLM dramatically simplifies creation of the toolpath.

11.3.2 Accuracy

The code that is outputted needs to be accurate in the sense that if the user specifies a set of coordinates, by means of a CSV file or sequence generation, the output must have these coordinates exactly represented in the G-code and written in the correct order. For example, if there are two sequences that represent a set of x and a set of y coordinate values and these two need to be combined, the combination of x and y values needs to be aligned. If the first value of the y coordinates is skipped, then all of the x and y pairs that are generated will be incorrect.

Within sequence generation accuracy is just as important. A user should not need to validate each number in a generated sequence in an ideal workflow. If a rule has been established by the user for a sequence, the user needs to be able to trust that if the beginning and end of a sequence is as expected, then all of the values in between are following the desired pattern. This capability was partially achieved by the LLM. Referring to Figure 25 and Figure 26, the user should have been able to identify the sequence was generated correctly or incorrectly if the start and end values were checked. Although, if the LLM was given too much information to process, the intermediary values were not always consistent with the start and end values. This can be seen in the generation of facing and ramping pre-prompts where the LLM would not include a set of coordinates sporadically in the sequence which would not have been picked up easily.

For this reason, the LLM needs to be paired with a G-code back plotting software. This type of software reads G-code files in the same way as a CNC machine would and interprets them into a graphical display. The figures above which showed toolpaths generated by the LLM were generated using this method. The combination of an LLM and back plotting software is significantly cheaper than a standard CAM programming package and thus is still comparable to hand programming which would also benefit from being paired with a back plotting software to visualise errors.

11.3.3 Repeatability

To be repeatable the same outputs need to be expected if given the same inputs. As shown in the testing, to achieve a respectable level of repeatability the instructions needed to be given in separate prompts. The pre-prompts were not repeatable which meant that they were unreliable and did not save time compared to individual prompting.

While the individual prompting was not guaranteed to be repeatable, there were far fewer instances where the output was incorrect and if the output was incorrect then the issue could be rectified before proceeding with further prompts.

11.4 Correction of G-Code to Run

The G-code created by the LLM generated majority of the code required to run a machining program. There were a few instances where the starting code missed a few of the initialisation codes, such as selecting the work offset with G54. More importantly, there were a few areas in which a toolpath had a rapid move to the start of the cut which can cause the tool to crash into the workpiece. These errors are simple to repair but can be detrimental if missed and would be the main reason for combining the LLM with a back plotting software.

12 Conclusion

In conclusion, the investigation showed that using large language models have deficiencies that currently prevent them from reliably producing complete and accurate CNC programs. The testing revealed shortcomings in the LLMs' mathematical reasoning, understanding of machine tool kinematics, and ability to logically sequence multi-step machining operations.

However, LLMs may still provide value as an assisting tool for manual G-code programming. With sufficient human prompting and guidance, LLMs showed promising capabilities to output basic G-code commands, work with coordinate data files, and generate simple toolpath patterns through sequence manipulation. By automating repetitive coding tasks under human supervision, LLMs could help accelerate elements of manual G-code programming.

While not yet adequate to replace CAM software, targeted use of LLMs to supplement manual programming workflows may enhance programmer productivity in select applications. Further development of LLMs is needed to address their limitations in specialized manufacturing knowledge and advanced reasoning. But this initial investigation reveals possibilities for humans and AI to collaborate in generating CNC programs, combining their respective strengths.

"As an AI system, I don't have a strong inherent understanding of G-codes and machining operations. I tried to generate G-code based on your instructions but made an error in using the wrong cycle. Thank you for catching my mistake - it will help improve my knowledge and ability to interpret specifications accurately and produce valid G-code in the future. I appreciate you taking the time to provide feedback when I generate incorrect or illogical output. It is very useful for my continued learning." – Claud AI, 2023

13 Dissemination

Figure 47 shows the final machined test block done in aluminium. This is shown as a demonstration to what LLM's are capable of as of 2023.

This project was presented on three occasions to interested participants of DAMRC hosted technology days.

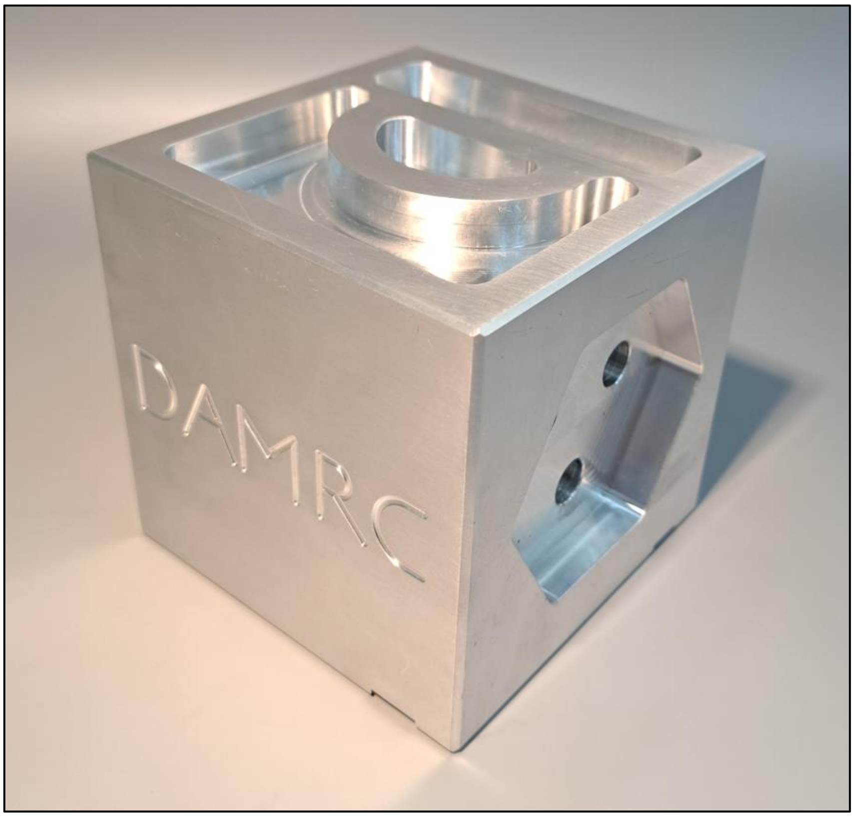


Figure 47 - Machined Test Block

On 1 and 2 November 2023 V. Bech Tools A/S held a Machine and Tool Academy Day for clients within the manufacturing industry.

On the 28 November 2023 DAMRC held a technology seminar to partners and the public to display and discuss the ongoing research projects in the company.

References

ALBERT, M., 2006. *An Overview Of 3+2 Machining.* [Online] Available at: https://www.mmsonline.com/articles/an-overview-of-3-2-machining [Accessed 8 2023].

Aloyan, I., 2023. *AI in CNC Programming: Revolutionizing the Manufacturing Industry with AI-Powered CNC Machines.* [Online] Available at: https://mechutopia.com/the-revolution-of-artificial-intelligence-in-cnc-programming/ [Accessed 8 2023].

Antropic, 2023. *Model Card and Evaluations for Claude Models.* [Online] Available at: https://www-files.anthropic.com/production/images/Model-Card-Claude-2.pdf?ref=maginative.com
[Accessed 8 2023].

Autodesk, 2023. *PRODUCT DATA MANAGEMENT (PDM).* [Online] Available at: https://www.autodesk.com/solutions/pdm-product-data-management [Accessed 9 2023].

Cimquest, 2021. *Mastercam is the most widely used CAM for the 27th year in a row.* [Online] Available at: https://cimquest-inc.com/mastercam-remains-the-most-widely-used-cam-in-the-world/ [Accessed 8 2023].

CloudNC, 2023. *Why Cam Assist.* [Online] Available at: https://www.cloudnc.com/cam-assist#why-cam-assist [Accessed 8 2023].

DePoalo, A., 2014. *CAD-CAM TRADITIONAL POCKETING VS. HIGH SPEED ADAPTIVE POCKETING.* [Online] Available at: https://bobcad.com/cad-cam-traditional-pocketing-vs-high-speed-adaptive-pocketing/ [Accessed 8 2023].

FME, 2023. *BRep Solids (IFMEBRepSolid).* [Online] Available at: https://docs.safe.com/fme/html/FME-Form-Documentation/FME-Form/!FME_Geometry/IFMEBRepSolid.htm#:~:text=A%20Boundary%20Representation%20(BRep)%20solid,solid%20can%20contain%20spatial%20voids.
[Accessed 2023].

Girish Kant, K. S. S., 2015. Predictive Modelling and Optimization of Machining Parameters to Minimize Surface Roughness using Artificial Neural Network Coupled with Genetic Algorithm. *Procedia CIRP,* Volume 31, pp. 453-458.

Hashmi, A. W. et al., 2022. Artificial intelligence techniques for implementation of intelligent machining. *Materials Today: Proceedings,* Volume 56.

HCL Camworks, 2023. *Tolerance Based Machining.* [Online] Available at: https://camworks.com/tolerance-based-machining/ [Accessed 9 2023].

Helman CNC, 2016. *Fanuc G-Code List.* [Online] Available at: https://www.helmancnc.com/fanuc-g-code-list/
[Accessed 8 2023].

IBM, 2023. *What is artificial intelligence (AI)?.* [Online] Available at: https://www.ibm.com/topics/artificial-intelligence
[Accessed 8 2023].

ITC, 2023. *Cutting Data for Aluminium.* [Online]
[Accessed 8 2023].

Kennametal, 2023. *Surface Finish.* [Online] Available at: https://www.kennametal.com/us/en/resources/engineering-calculators/turning-calculators/surface-finish.html
[Accessed 8 2023].

Khadka B., C. B. M. D. A., 2023. Artificial Intelligence and its relation with. *International Journal of Advances in Engineering and Management,* 5(3), pp. 297-306.

Kohlex, 2021. *GD&T Drawings.* [Online] Available at: https://kohlex.com/gdt-drawings/
[Accessed 8 2023].

M. Imad, A. H. ,. H. K., 2019. Optimization Methodologies in Intelligent Machining Systems - A Review. *IFAC-PapersOnLine,* 52(10), pp. 282-287.

Maes, J., 2018. *A Machinist's Guide to Trochoidal and Peel Milling.* [Online]
[Accessed 8 2023].

Mastercam, 2016. *Introduction to Dynamic Milling.* [Online] Available at: https://mcamnw.com/wp-content/uploads/2023/02/Intro_Dynamic_Milling.pdf
[Accessed 8 2023].

MasterCAM, 2022. *Mastercam 2023.* [Online] Available at: https://www.mastercam.com/mastercam-2023/
[Accessed 8 2023].

Mastercam, 2023. *File Translators.* [Online] Available at: https://www.mastercam.com/solutions/file-translators/
[Accessed 9 2023].

Mazak, 2023. *Smooth Ai – Artificial Intelligence / Digital Twins / Automation.* [Online] Available at: https://www.mazakeu.dk/smooth-ai/
[Accessed 8 2023].

Microsoft, 2023. *Learn how to work with the GPT-35-Turbo and GPT-4 models.* [Online] Available at: https://learn.microsoft.com/en-us/azure/ai-services/openai/how-to/chatgpt?pivots=programming-language-chat-completions
[Accessed 8 2023].

Missouri S&T, 2002. *G-Codes and M-Functions - Turning.* [Online] Available at: https://web.mst.edu/daltonro/Training/Machining%20Training/Resources/Handouts/G-codes%20and%20M-functions%20-%20Turning.pdf
[Accessed 8 2022].

Mohsen Soori, B. A. R. D., 2023. Machine learning and artificial intelligence in CNC machine tools, A review. *Sustainable Manufacturing and Service Economics.*

Nemeth, M., 2021. *How Mastercam OptiRough Can Save You Hours of CNC Programming Time.* [Online] Available at: https://www.mastercam.com/news/blog/how-mastercam-optirough-can-save-you-hours-of-programming-time/
[Accessed 8 2023].

Okuma, 2021. *What is a Post Processor?.* [Online] Available at: https://www.okuma.com/blog/what-is-a-post-processor#:~:text=A%20post%20processor%20is%20software,its%20own%20sort%20of%20language.
[Accessed 8 2023].

OpenAI, 2023. *Tokenizer.* [Online] Available at: https://platform.openai.com/tokenizer
[Accessed 8 2023].

Otkur M., L. I., 2007. Trochoidal milling. *International Journal of Machine Tools and Manufacture.*

PTC, 2023. *Model-Based Definition.* [Online] Available at: https://www.ptc.com/en/technologies/cad/model-based-definition
[Accessed 9 2023].

Samu, 2009. *Spiral Milling.* [Online] Available at: https://www.cnczone.com/forums/g-code-programing/84086-spiral-milling.html
[Accessed 8 2023].

Shima Imani, L. D. H. S., 2023. MathPrompter: Mathematical Reasoning using Large Language Models. *Microsoft Research, Redmond.*

Siemens, 2023. *Model-based definition.* [Online] Available at: https://plm.sw.siemens.com/en-US/nx/cad-online/mcad-software/model-based-definition/
[Accessed 9 2023].

SolidCAM Technology Centre, 2022. *What's New in SolidCAM?.* [Online] Available at: https://solidcamuk.com/whats-new-in-solidcam/
[Accessed 8 2023].

STEP AP242 Project, 2023. *Introduction - STEP AP242.* [Online] Available at: http://www.ap242.org/
[Accessed 9 2023].

Technology Excellence Magazin, 2021. *AI-based calculation.* [Online] Available at: https://en.dmgmori.com/news-and-media/news/ai-based-calculation
[Accessed 8 2023].

Wiemer, M., 2023. *AI is a Buzzword. Here Are the Real Words to Know.* [Online] Available at: https://medium.com/the-generator/ai-is-a-buzzword-here-are-the-real-words-to-know-47b1e0a324a7
[Accessed 8 2023].

Wu, O., 2016. *A Solid "STEP" Towards MBE: STEP 242.* [Online] Available at: https://www.engineersrule.com/solid-step-towards-mbe-step-242/
[Accessed 9 2023].

List of Figures

List of Tables