PAPER • OPEN ACCESS

Interpreting the G-code of drilling machining to use in open CNC controller machine

View the article online for updates and enhancements.

You may also like

Interpreting the G-code of drilling machining to use in open CNC controller machine

Noor Hatem[1,3], Yusri Yusof[1], Aini Zuhra A Kadir[2], Kamran Latif[4], Mohammed M A[2,5]

[1] Faculty of Mechanical and Manufacturing Engineering, University Tun Hussein Onn Malaysia
[2] Faculty of Mechanical Engineering, University Technology Malaysia
[3] Department of Petroleum Engineering, College of Engineering, University of Basrah
[4] Faculty of Mechanical and Manufacturing Engineering Technology, Universiti Teknikal Malaysia Melaka (UTeM), Malaysia
[5] Department of Materials Engineering, College of Engineering, University of Basrah

E-mail: yusri@uthm.edu.my, kamran@utem.edu.my

Abstract. Computer numerical control (CNC) software is the "brain" of a machine tool as it controls the machine tool's movement. The development of open CNC controller systems is one of the most popular topics in the last three decades due to the current CNC remains exclusive and costly. However, there is no open-source system that can use with an open CNC controller. In this developed system, the drilling G-code was analyzed to extract the points before simulating and sending it to any open CNC controller machine. Among the machining processes used most extensively is drilling. The used modules and functions are created by LabVIEW. The results show the system's extracted points are similar to the draw drilling point in Solid work.

Keywords: Open CNC controller, Drilling, G-code and LabVIEW.

1. Introduction

Technological advances are continuously lowering the cost of CNCs so that small enterprises or individuals can now undertake their assembly. These machines are compatible with a wide range of applications. Their advantages are because they are much more accurate than manual work, speed up productions, and enable mass-scale productions. Furthermore, they are not difficult to maneuver as long as their programming is correct [1]. Moreover, in terms of development, CNCs are very promising, and a large number of machining processes can be performed based on basic operations like translation, rotation, and regulation of CNC head status. In fact, the algorithms underpinning traditional processes are largely the same as those employed for the programming of cutting-edge devices (e.g. 3D printers) [2]. Over the past few years, the machine device's traditional controller has been increasingly undermined by the PC-based open CNC system that is not dependent on the CNC merchant and can execute client-characterized application programs [3]. Furthermore, a personal computer can be considered hardware that functions using real-time operating system software, thus decreasing costs while increasing the control system's flexibility. Developing an open architecture platform can make the

CNC application framework more flexible as well as modular. It will also allow the system users to customize the designed function according to the demand concerning different applications. Thus, changing the programming or coding can make the software reusable. This enhances the overall system's performance as it simply upgrades the hardware system [4]. The past 30 years have been increasingly focused on the subject of developing open CNC systems. There have been several open CNC prototypes, including Open System Architecture for Control within Automation Systems (OSACA) [5], Open Modular Architecture Controller(OMAC) [6], LinuxCNC [7], TwinCAT [8], and Open System Environment Consortium (OSEC) [9].

In order to overcome these issues, a new ISO data interface model (ISO 14649) was introduced in 1998 [10]. While the implementation of G-code was found to be vendor-dependent in commercial CNC systems. In order to overcome that issue, the researchers used software development to produce open CNC systems [4]. Basically, the open CNC systems used limited features such as interpretation and simulation of the input code data.

2. The architecture of the optimized system

The theoretical framework of the developed system focused on interpretation and simulation functions. The input data were extracted and further verified by graphical simulation. These modules were designed in a virtual component technology-based tool (NI LabVIEW).

The system started by reading the input file before extracts code points. The *information* module provides the path control to upload files, read the input file's complete contents, and help the interpreter with the input file's starting point. The module includes the *case route*, *recite*, along with *guide route* functions modules, as shown in Figure 1. Moreover, the *case route* functional module provides the input file with a path control to upload using LabVIEW "File Path Control" function, whereas the *recite* functional module reads the input file's complete contents using "Read from Text File Function". Further, the *guide route* functional module guides the interpreter with the interpretation's starting point and enables the code to be read line by line using "Pick Line Function" at the LabVIEW functions library.
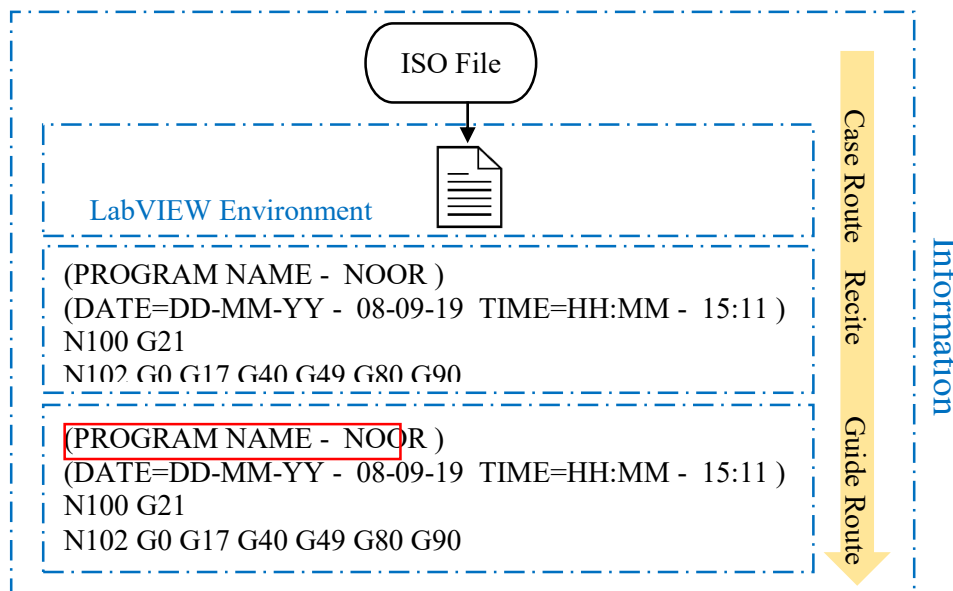

Figure 1. Information module for G-code and STEP-NC with the mechanism

The *extraction* module's function is used to extract information from ISO data interface models regarding the axis position, feed rate, spindle, as well as tool. As shown in figure 2, this module uses different functional modules, including *match*, s*plit*, *excerpt*, and *token*. The *match* functional module is in charge of identifying entity numbers in input code using "Match Regular Expression Function".

For G-code, the line started with N letter only selected. The s*plit* functional module then determines the entity numbers' matched data using "Search/Split String Function". This function is looking for "X", "Y", "Z", "F", "S" and "T" letters to find the related values. Next, the *excerpt* functional module extracts the data from the matched information. Finally, the *token* functional module distributes this extracted data into different sections.
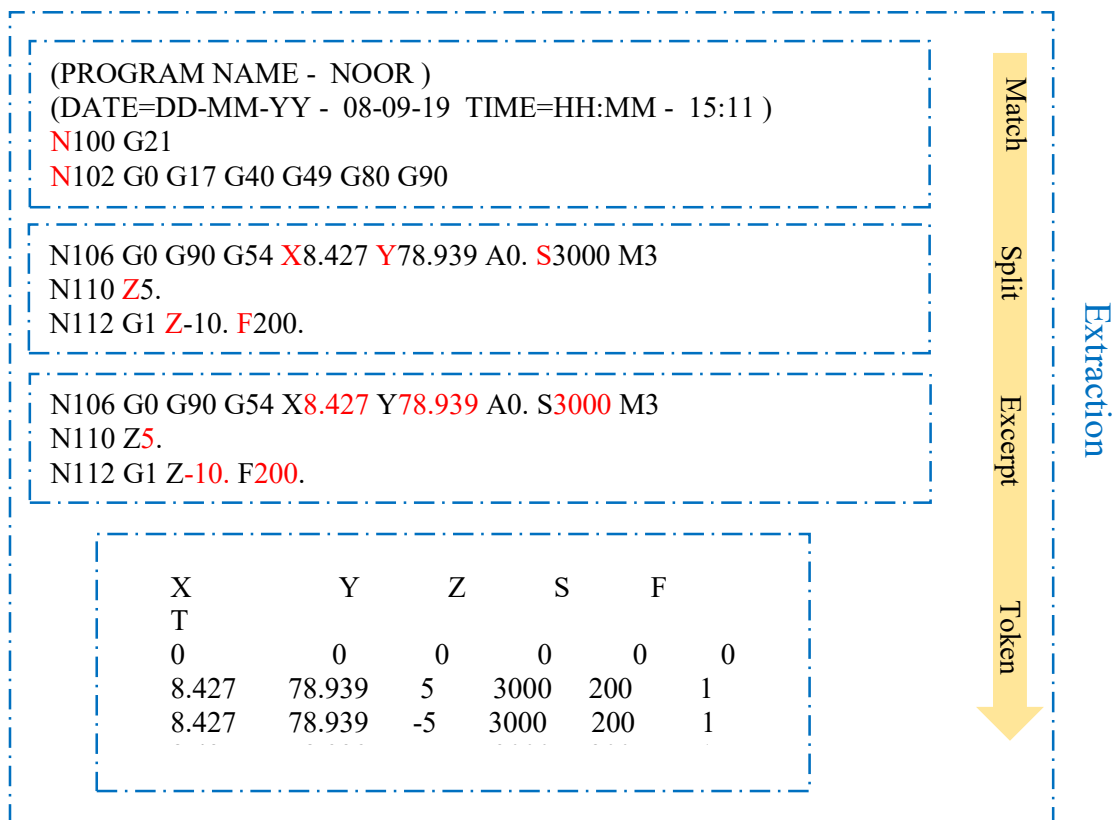

Figure 2. Extraction module for G-code and STEP-NC with the mechanism

3. Simulation Module

The function of this module is to graphically verify the ISO and optimized data code before performing real machining operations. This module is composed of *recite* and *plotting* functional blocks. The *recite* functional module is responsible for selecting the point from the array that is the opposite of the index of ant path and saving the data into a new path points array. The *plotting* functional module is responsible for taking the axis positions (x, y, and z) data from the *recite* functional module and plotting them as 3D graphical forms.

4. Case Study

The validation of this system was performed through testing on tool paths of 9 holes. Figure 3 shows the CAD drawing using "Solid Work" software. The G-code data generated from this drawing and

extracted. Figure 4 shows the result of the system application of the *information* and *extraction* module in the LabVIEW interface.
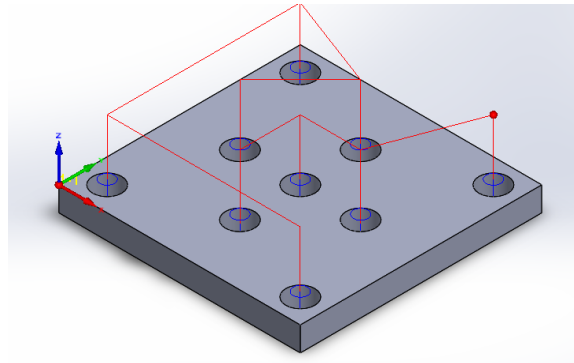

Figure 3. The drawing example

(a)

(b)

Figure 4. The system interface shows the result from (a) the Information module and (b) the Extraction module

The system read the file before extracting the points from the input code. Figure 5 shows the extracted points graphically. As shown in figure (5), the system extracted points were successfully, and the points can be used in any open CNC controller systems.
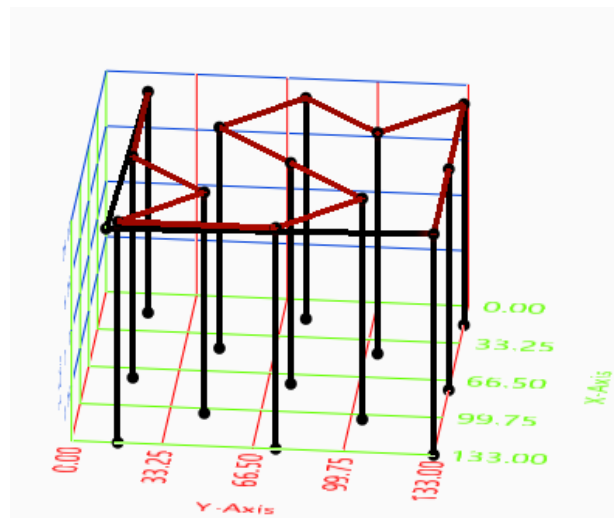
Figure 5. Simulate the extracted points

5. Conclusion

The present study developed a new system to use with an open CNC controller. The developed system is used for drilling machining. This system used a multi-module to read, extract, and simulate the data from the input G-code. The developed system was performed via the LabVIEW software with new functional blocks. The system is open source and can be modified to use with another machining process such as milling, turning or both.

References

[1]   P. SMID, "CNC Programming Handbook–Ebook." New York: Industrial Press Inc, 2005.
[2]   J. F. Kelly, *3D printing: build your own 3D printer and print your own 3D objects*. Que Publishing, 2013.
[3]   M. Paprocki, A. Wawrzak, K. Erwiński, K. Karwowski, and M. Kłosowiak, "PC-based CNC machine control system with LinuxCNC software," *Meas. Autom. Monit.*, vol. 63, no. 1, pp. 15–19, 2017.
[4]   L. Liu, Y. Yao, and J. Li, "A review of the application of component-based software development in open CNC systems," *Int. J. Adv. Manuf. Technol.*, pp. 1–27, 2020, doi: 10.1007/s00170-020-05258-1.
[5]   P. Lutz, W. Sperling, D. Fichtner, and R. Mackay, "OSACA—The vendor neutral control architecture," in *Proc. European Conf. Integration in Manufacturing*, 1997, pp. 247–256.
[6]   J. L. Michaloski, S. Birla, G. F. Weinert, and C. J. Yen, "Framework for component-based CNC machines," in *Sensors and Controls for Intelligent Machining, Agile Manufacturing, and Mechatronics*, 1998, vol. 3518, pp. 132–143.
[7]   "LinuxCNC," *LinuxCNC*, 2018. http://linuxcnc.org/docs/2.7/ html/getting-started/about-linuxcnc.html.
[8]   Beckhoff, "PC-based control," *Beckhoff*, 2020.

[9]   C. Sawada and O. Akira, "Open controller architecture OSEC-II: architecture overview and prototype systems," in *1997 IEEE 6th International Conference on Emerging Technologies and Factory Automation Proceedings, EFTA'97*, 1997, pp. 543–550.

[10]   S.-H. Suh and J.-J. Lee, "Five-axis part machining with three-axis CNC machine and indexing table," *J. Manuf. Sci. Eng.*, vol. 120, no. 1, pp. 120–128, 1998.