Int J Adv Manuf Technol (2017) 89:1013–1024
DOI 10.1007/s00170-016-9124-0

ORIGINAL ARTICLE

Development of a feature-based open soft-CNC system

Kamran Latif[1] · Yusri Yusof[2] · Aydin Nassehi[3] · Qadir Bux Alias Imran Latif[4]

Received: 5 January 2016 / Accepted: 29 June 2016 / Published online: 16 July 2016

Abstract Modern computer numerical control (CNC) is intended to be more flexible, interoperable, adoptable, open, and intelligent. In realisation of such a CNC system, an ISO standard known as STEP for numeric control (STEP-NC) or ISO 14649 was developed in 2004 to alleviate a number of challenges associated with the widely used CNC standard (ISO 6983). Implementation of STEP-NC was initially carried out on some commercial CNC systems via an indirect STEP-NC programming approach. However, this approach failed to meet all the requirements of a modern CNC system, due to the translation of data from the high level to low level, which is vendor-specific and machine tool dependent, thus detrimental to the basic resource independent philosophy of STEP-NC. This paper presents a new generation of CNC systems, based on the Open Architecture Control (OAC) technology supported by STEP-NC data model to fundamentally solve these challenges. The developed system employs a new set of techniques for STEP-NC data interpretation, graphical verification, execution, monitoring, and report generation that supersede the exiting techniques in scope and capability. Using these techniques, the proposed system provides a flexible, intelligent, and adaptable manufacturing platform that can provide unprecedented levels of scalability and resource allocation agility in modern manufacturing enterprises. A prototype implementation of the proposed model is based on STEP-NC interpretation, 3D simulation, machine motion control, and live video monitoring with automatic document generation modules has been used to verify and validate the system in conjunction with industrially inspired test components.

Keywords CNC · STEP-NC · Interpreter · Simulation · Monitoring · Open architecture controller

✉ Kamran Latif
kamran@dhu.edu.my; engr.kamranqureshi@gmail.com

Yusri Yusof
yusri@uthm.edu.my

Aydin Nassehi
a.nassehi@bath.ac.uk

Qadir Bux Alias Imran Latif
imranqazi37@gmail.com

[1] Faculty of Engineering, DRB-HICOM University of Automotive Malaysia, Pekan, Malaysia

[2] Faculty of Mechanical and Manufacturing Engineering, University Tun Hussein Onn Malaysia, Parit Raja, Malaysia

[3] Department of Mechanical Engineering, University of Bath, Bath, England

[4] Department of Civil and Environmental Engineering, College of Engineering, University of Nizwa, Nizwa, Oman

1 Introduction

Computer numerical control (CNC) machines have been in operation for more than half century, mainly using a 50-year-old language ISO 6983, also known as G-code. G-code is a low-level language, mainly specifying the machine axes motions in terms of position and feed rate. There are a number of well recognised problems with this data model, including limited bandwidth, unidirectional information flow from CAD/CAM to CNC, absence of integrated CAD-CAM-CNC, and difficulties to make last minute changes at

shop floor [1]. This standard is therefore, not able to fulfil the requirements of the modern manufacturing environment.

In order to overcome these problems, the International Standards Organization (ISO) introduced a new ISO standard for NC data model, namely, ISO 14649 or otherwise known as STEP-NC [2]. The objective of STEP-NC is to integrate design data (in the format of STEP [3, 4]) with CAM and CNC (in the format of STEP-NC). This is possible as STEP-NC and STEP use the same data modeling and presentation methods. STEP-NC therefore supports "Design anywhere, builds anywhere, and supports anywhere" [5]. The introduction of STEP-NC prevents information loss in the CAD-CAM-CNC data chain and opens the doors for the development of next generation (modern and flexible) CNC systems [6]. The implementation of STEP-NC was first carried out on some of the commercially available CNC systems via an "In-Direct" STEP-NC programming approach [7, 8]. This approach translated the STEP-NC data into G-codes that were subsequently fed to a CNC controller for execution [6]. Indeed, this approach was not enough to enable all the features of STEP-NC [6]. This is mainly due to the fact that current commercial CNC machines are mostly of a close nature [9]. Subsequently, a new STEP-NC implementation approach, named "Interpreted STEP-NC programming" [7, 8], was introduced. In this approach, the controller directly reads and interprets the STEP-NC information as per internal structure of the machine.

Open architecture control (OAC) technology was introduced into CNC systems in order to tackle the issues of the closed nature of CNC machines. The aim of OAC was to develop controllers that are independent on manufacturer's proprietary technologies, allowing the user to use hardware and software from several different manufacturers and freely assemble the acquired pieces of equipment [10]. Personal computer (PC) has been considered as one of the preferred hardware platforms for open CNC systems, because of its good openness, low cost, and high performance to price ratio [11], as supported by Park and his research team [12]. Ma, along with other researchers have also supported the sentiment that the current trend of CNC system development must be towards the PC-based soft-CNC systems [13]. A number of systems have been developed, based on the OAC technology [8, 14–18] and [19]. These systems were developed by using general purpose object-oriented programming languages such as C, C++, JAVA, VB, and others. Meanwhile, Wang and Minhat introduced a new method of STEP-compliant system development, based on functional blocks and OAC technologies [20, 21]. However, the system was based on the indirect STEP-NC programming approach. This paper reports on a new technique for the development of a STEP compliant open CNC system which is based on Interpreted STEP-NC programming approach. The proposed technique is based on the virtual component technology. The developed system can interpret an ISO 14649-21 physical file, graphically verify the translated code before actual machining operations, provide live monitoring of machining operations and automatically generate the documentation of the performed job in the MS word and PDF formats.

In the past, researchers utilized virtual component technology for STEP-based open CNC system development [22–25], but their work was mostly limited only to a three-axis machine in an open loop control system environment. In comparison to those approaches, the proposed machine motion control technique is able to control three-axes motions, spindle, and automatic tool changer motions in a closed loop control system. Also, this technique introduced a new way of STEP-NC part 21 physical file translation, graphical verification, process monitoring, and document generation by utilizing virtual component technology. The developed system is composed of the *STEP-NC interpretation*, *3D simulation*, *machine motion control*, and *live video monitoring with automatic document generation* modules. The rest of the paper is organised to include rage architecture of the system, algorithm design of the system, GUI of the system, experimental study, and conclusions.

2 Architecture of the system

The system is designed to execute STEP-NC data interface model part program in open architecture control-based CNC system. The system architecture is based on the virtual component technology. The design of the system integrates the STEP-NC, open architecture control, and virtual component technology into a single unit and builds a new CNC cell control system. The theoretical framework of the developed system focused on four major features namely: interpretation, simulation, machine motion control, and modern functions. In the first feature, the development of translator has been carried out to interpret STEP-NC part 21 physical file. This translation output is further verified by graphical simulation in the second feature. To do so, a virtual component technology-based module has been developed. Once the translation output is verified, the real-time machining need to be carried out. The third feature focused on the development of real-time machine motion control module. The last feature focused on the enabling for modern functionalities into the CNC core, where in this system, the process monitoring and automatic document generation functionalities have been highlighted. All these four features have been achieved by four different modules. These modules have been designed in virtual component technology-based tool (NI LabVIEW). This technology can enable access to monitoring, inspection, database, Internet

connectivity, and other functions within a single platform. In other words, it can be a useful source to provide an all-in-one platform for CNC machines. These facilities can make LabVIEW a very useful tool in the development of modern CNC systems. Overall, the architecture of the developed system is composed of the following four modules: *STEP-NC interpretation*, *3D simulation*, *machine motion control*, and *live video monitoring with automatic document generation*, which are further composed of various sub modules that contains numerous functional blocks as shown in Fig. 1.

2.1 STEP-NC interpretation module

The function of this module is to translate STEP-NC data interface model information as per required structure of CNC machine. It interprets the acceleration, deceleration, spindle speed, feed rate, tool, tolerance, and other information from ISO 14649 code, so that the machine can be moved throughout linear or circular interpolation and perform operations. This module provides shop floor editing facilities to the CNC system and is also able to generate physical file output in user defined structure of .txt and .xml formats. The internal structure of this module is composed of three sub modules: *participation*, *mining*, and *production*, which are further composed of various functional blocks as shown in Fig. 1.

2.1.1 Participation sub module

The purpose of this sub module is to provide the path control for file uploading, to read the complete contents of input file and to guide the interpreter regarding the starting point of the input file. This module is composed of the *input path control*, *read data*, and *line indexing* functions blocks. The *input path control* functional block is responsible to provide a path control for input file uploading. The *read data* functional block is designed to read the complete contents of input file while *line indexing* functional block is intended to guide the interpreter regarding the starting point of the interpretation. The *line indexing* function block enables the functionality of reading the code from any line of the input file. By default, this function block has line index value equal to zero that indicates the first line of the input code.

2.1.2 Mining sub module

The function of this sub module is to extract the information of axis position, spindle, feed rate, tool, and tolerance from STEP-NC data interface model by utilizing various functional blocks such as; *search*, *match*, *extract*, *token*, *store*, and *combine* as shown in Fig. 1. The *search* functional block searches for the entity numbers in STEP-NC input code. In STEP-NC part 21, the starting point is titled with "PROJECT" string, therefore, the functional block

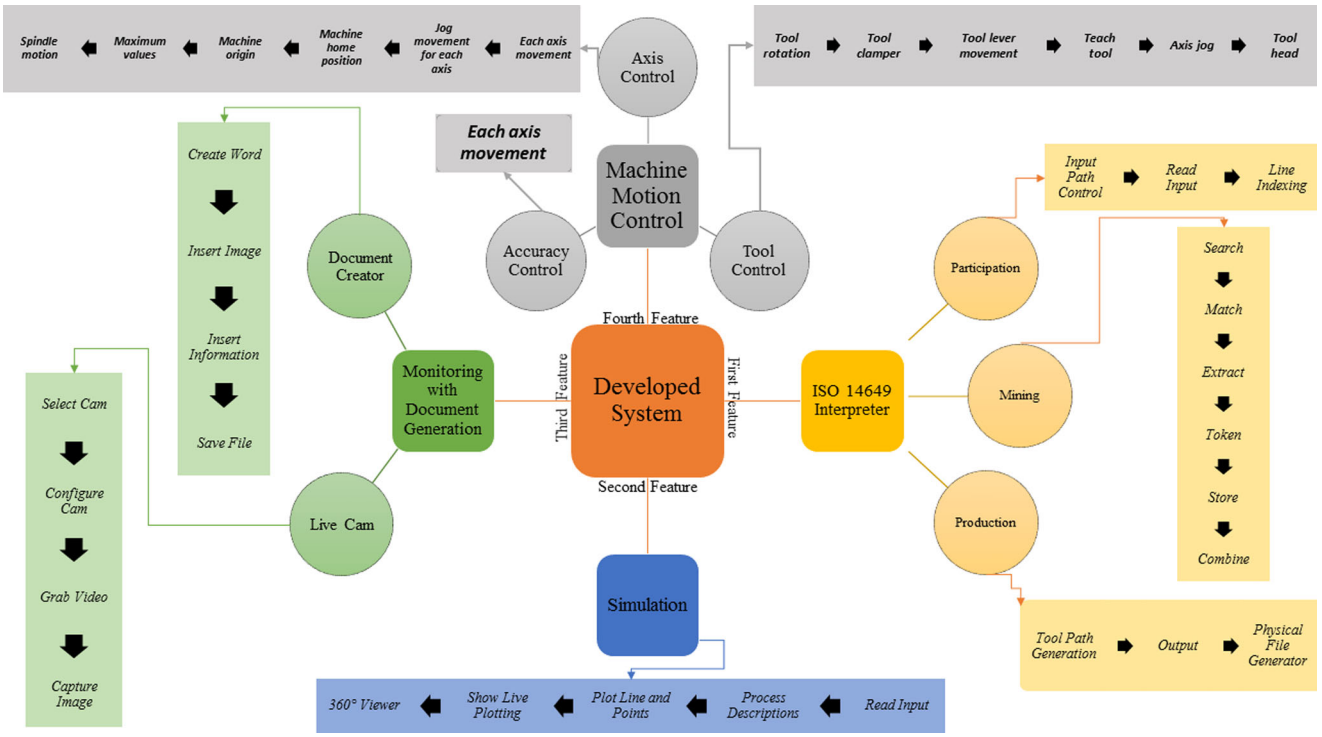

Fig. 1 Architecture of developed system

first searches for that string inside "DATA" section. Subsequently, the function starts searching for the entity numbers (i-e; #2, #4) inside input file. The *match* functional block is responsible to find the matched data of the entity numbers (#2=WORKPLAN ('MAIN WORKPLAN',(#54,#55),#60). The *extract* functional block is responsible to excerpt the data from matched information. In case of STEP-NC, this sub module extracts the entity names, entity number, and values data as per process type. The *token* functional block is responsible to distribute the extracted data into separated sections. For example, entity names are in one section, entity numbers are in another section and values are in yet another section. The *store* functional block is responsible to save all these sections in the shape of arrays. Finally, the *combine* functional block combines the (names and values) data into a new single code and passes it to the *production* sub module. Figure 2 shows the functionality of this sub module.

This sub module is the main phase of the *STEP-NC interpretation* module, where it provides all the data of the input STEP-NC part 21 code into different sections for easy understanding. This sub module extracts the working steps, machining working steps, setup, tool, work piece, tolerance, locations, depths, directions, and other data from the STEP-NC code. This sub module also provides the editing functions to all the extracted data in off line and as well as on line mode. In other words, this sub module enables the shop floor editing function in the developed system.

2.1.3 Production sub module

The function of this sub module is to pass the interpreted data to the CNC machine and also able to generate the output physical file in user defined hoc structure of .txt and .xml formats. In addition, this sub module also generates the tool paths from the STEP-NC interpreted data. This is important because ISO 14649 code does not contains tool paths, rather, it only contains the process starting point coordinates values and length measurement values. It is the aim of STEP-based CNC systems that the controller should generate the tool paths by itself. This sub module is composed of *tool path generator*, *output*, and *physical file generator* functional blocks. The *tool path generator* functional block is responsible to automatically generate the tool paths from translated STEP-NC information. The *output* functional block is responsible to parse the output of the mining sub module to the hardware parts of the CNC machine. The *physical file generator* functional block is responsible to create the physical file output of the translated data in .txt and .xml formats as per user defined hoc structure.

2.2 3D simulation module

The function of this module is to graphically verify the *STEP-NC interpretation* module translated code before performing real machining operations. This module is composed of *read input*, *process description*, *plot lines and points*, *show live plotting*, and 360° *viewer* functional blocks as shown in Fig. 1. The *read input* functional block is responsible to read the input data. It acquires the output of *tool path generator* functional block of the *STEP-NC interpretation* module as an input. This functional block first reads the complete contents of input data and then passes this information to the *process description* functional block that extracts the process names data from the input code. The *plot lines and points* functional block is responsible to take the axis positions (x, y, and z) data from *read input*

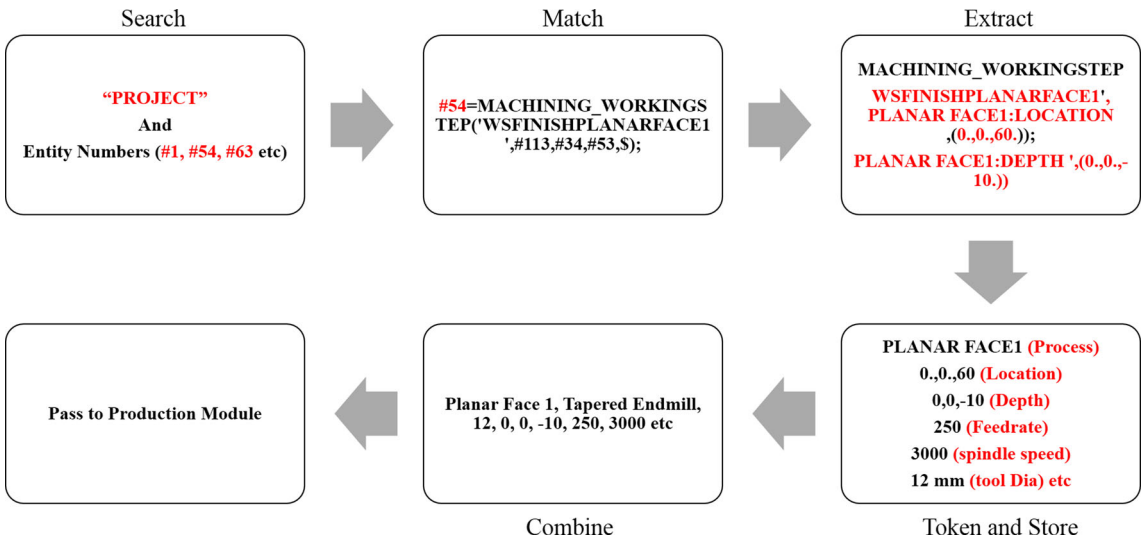

Fig. 2 Functionality of mining sub module

functional block and plot it into graphical form in 3D. The *show live plotting* functional block applies delay operations in plot lines and points functional block to show the live plotting of input data. The 360° *viewer* functional block is responsible to enable the rotational views into the module and show the plotting in various angles.

2.3 Machine motion control module

This module is responsible for controlling the motion of three-axis CNC milling machine. In this system, a three-axis motion control technique of Elias and his research team were utilized for the motion controlling of X, Y, Z axis and spindle of the CNC milling machine [25, 26]. The technique was limited to three-axis and spindle controls in an open loop control environment only. Whereas, in this study, the technique was extended to three-axis motion control with automatic tool changer in a closed loop control environment. Overall, the development of this module was divided into two configurations: hardware and software. The hardware configuration was based on the interaction of CNC machine with PC. In this study, a three-axis Den ford NOVAMILL was converted into a PC-based CNC system to make an open CNC environment. This installation makes the CNC controller free from vendor specifications and shows a minute part of sustainable manufacturing.

The software configurations of this module are composed of three sub modules: *axis control*, *tool control*, and *accuracy control*, which contain various functional blocks as shown in Fig. 1. The *axis control* sub module contains functional blocks for *each axis movement*, *jog movement for each axis*, *machine home position*, *machine origin*, *maximum values*, and *spindle motion*. The *axis movement* functional block controls the motion of three-axis CNC machine in X, Y, and Z coordinates. The *jog movement* functional block sets the jog movement of X, Y, and Z axes of CNC machine. The *machine home position* functional block is responsible to set and move the machine at its defined home position. Similarly, the *machine origin* functional block sets and moves the machine at a defined origin point. The *maximum values* functional block is responsible to define maximum range of machine axis, spindle, and tool changer movements to avoid accident. While the *spindle motion* functional block controls the motion of the spindle of the CNC machine.

The *tool control* sub module is composed of *tool rotation*, *tool clamper*, *tool lever movement*, *teach tool*, *axis jog*, and *tool head* functional blocks. The *tool rotation* functional block controls the rotations of the tool changer. The *tool clamper* functional block controls the clamping and un clamping of tool changer. The *tool lever movement* functional block is responsible to control the linear movement of the tool changer. The *teach tool* functional block tells the system regarding the changes made by the user while setting. The *axis jog* functional block is responsible to set the jog movement of the tool changer in rotational and linear movements. Lastly, the *tool head* functional block highlights the tool that is already installed in the clamper. The last sub module is of *accuracy control* that contains functional blocks for each axis of the CNC machine, which reads the actual position from rotatory encoders and shows the actual machine movement in real time. Figure 3 shows a simplified diagram of this module.

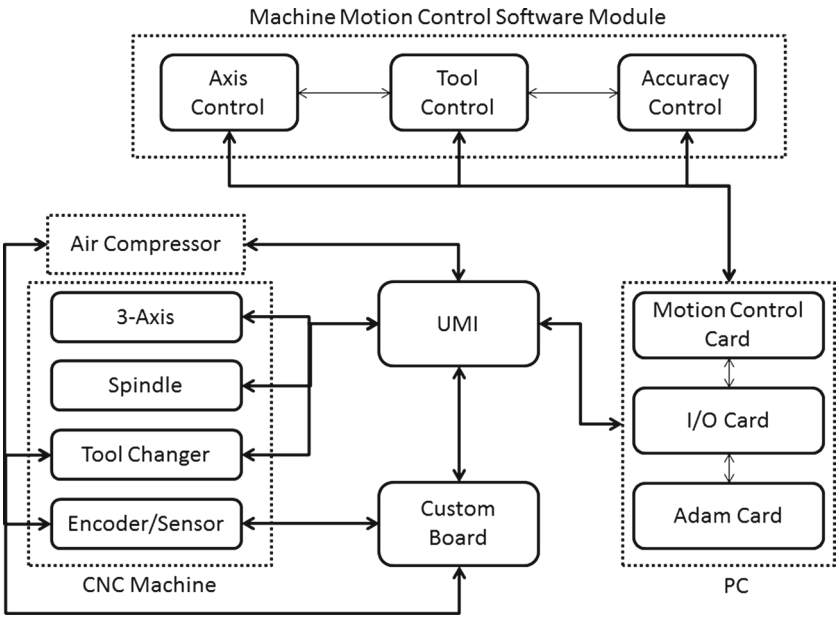

Fig. 3 Machine motion control module

2.4 Monitoring and documentation module

This module is responsible to display the live video of machining processes via camera and automatically generate report of performed machining operations. The purpose of live video monitoring is to enable process monitoring facilities into the system because in some cases, it is very hard to observe the machining process with naked eyes and it is considered to be a very dangerous act as well. In order to overcome this problem, *live video monitoring* sub module is designed and developed by using Image Acquisition (IMAQ) tool kit of NI LabVIEW and it is composed of various functional blocks namely: *select cam*, *configure cam*, *grab video*, and *capture image*. The *select cam* functional block searches for the installed camera devices into the system and shows them in the form of a list. The *configure cam* functional block is responsible to provide configuration options for camera resolution, color settings, zooming, and drawing markings in live video. The *grab video* functional block connects the camera device with software and acquires the live video. The *capture image* functional block is responsible to capture the pictures from video and save it into .jpg, .bmp, .tiff, and formats.

The other sub module of this module is the *automatic document generation*. The aim of this module is to automatically generate the report of performed machining operations. This module enables the CNC system to generate the soft document report that will minimize the use of papers which includes a minute glance of green manufacturing into the system. The module is composed of *create doc*, *insert image*, *insert information*, and *save file* functional blocks. The *create doc* functional block is responsible for the generation of blank Microsoft (MS) word file. The *insert image* functional block is responsible to insert image data into the generated MS word file. The *insert information* functional block is responsible to insert the text data into the generated MS word file and the *save file* functional block is responsible to save the generated file in .docx and .pdf formats into the system memory.

3 Algorithm design of the system

The developed system algorithm starts with the *input path control* functional block that provides the path to load STEP-NC part 21 file. Subsequently, the *read data* functional block will execute, reading the complete content of that input file. This function has a decision module that scans inside input file and makes decision regarding status of the file. If the input file does not contain any STEP-NC data, the decision module sends the signal towards error message. On the other hand, if it contains data, this module passes the file to the *line indexing* function block. The aim of this function is to provide access to read the input file from any point.

After the selection of *line index*, the data is forwarded to the *search* functional block. This functional block first searches for the "PROJECT" string before it proceeds with the search for entity numbers (#1, #2, etc) inside the input file. During first and second time readings, if the functional block could not find the "PROJECT" string or entity number, it will pass the signals to error message. In case it does, the found entity numbers are passed to the *match* functional block that reads the input file and searches for the entity number values. The output of this function is the complete data of the entity number for example #54 = MACHINING_WORKINGSTEP ('WSFINISH PLANAR FACE1', #113,#34, #53). Then, the matched data is entered to the *extract* functional block that firstly searches for the machining process and extracts the data as per process type. This functional block is linked to the process names library that guides the interpreter regarding the data mining structure. This is useful, because in the STEP-NC file, each process has some difference in series of information placements due to different amount of information within different processes. The *token* functional block will later execute that breaks and distributes the data into separate sections. The *token* functional block divides the data into separate arrays such as entity numbers array, process names array, locations values array, depth values array, and tolerance values array. At the end, *token* functional block passes all the information to the *store* functional block that saves all the data in its internal memory. After saving the data, the *store* functional block recalls the *search* functional block to search for entity numbers inside the stocked arrays. At this stage, the algorithm makes a decision. If the entity number is found, it recalls all the processes from *match* to *store* functional blocks. However, when no entity number is found in the stored data arrays, the functional block parses the data to the *combine* functional block that combines all the data into a new single code and passes it to the *production* sub module. The output of the *combine* functional block contains the information of process names, its location values, depth values, tolerance length measurement values, tool values, spindle speed values, and feed rate values.

The *production* sub module takes the combined code and passes it to the *tool path generator* functional block. It utilizes, all the information of the combined code and generates tool paths for the specified processes. This functional block is also connected to the process name library that guides regarding the selection of sub tool path generation blocks, according to specific process type (facing, drilling, pocket, etc.). Subsequent to the *tool path generation* functional block, the *output* functional block executes, that parses the information to the hardware parts of the CNC

machine in required structure. At the same time, the *production* sub module is able to generate the output physical file in .txt and .xml formats as per user defined structure with the help of the *physical file generator* functional block. Figure 4 shows functionality of *STEP-NC interpretation* module algorithm design.

After that, the *read input* functional module of *3D simulation* block takes the data from *STEP-NC interpretation* module and reads the complete contents of input file. Once the reading completes, the data is parsed to the *process description* functional block that extracts the process names from input code and parses the remaining data to the *plot points and lines* functional module which plots the graph of input data. In parallel to this functional block, the *show live plotting* functional block is also executed which shows the live plotting of the input data by applying delay functions. After plotting, the *360° viewer* functional block is executed that shows the plotting in various angles by rotating the plotting image in 3D environment. Then, the decision module starts that allows the user to make the decision regarding the simulation results. If the results are satisfactory, the module will parse the data to *machine motion control* module. If the results are not satisfactory, the module will parse the data back to *STEP-NC interpretation* module for modifications.

As the data is parsed to the *machine motion control* module, the machine starts operations with *home position* functional block that moves the machine at defined home position. With the start of machine operations, the *live cam* functional block is also activated in parallel that shows the live video of machining operations. At this position, the algorithm contains a decision module regarding home position settings by utilizing *home position* functional block. After home position setup, the *tool change* sub block is activated with *teach tool mapping* functional block for setting of cutting tools sequences, alignment, tool head, and others. After the tool changer settings are completed, the *load file* functional block is executed that provides a platform to upload translated STEP-NC data interface model information into the system. After that, the *start* functional block is executed that moves the machine at the origin position. At this moment, the algorithm contains a decision module regarding origin point setting by utilizing *origin position* functional block. After that, the machine execution is started as per translated ISO data interface model instructions by utilizing *each axis movement*, *spindle*, and *tool* functional blocks. When the time of tool change is due, the algorithm activates *tool control* sub module. This moves the machine back to home position, and subsequently searches for installed tool by utilizing *tool head functional* block. After tool at head learnings, the *tool rotation* functional block is executed that will first rotate the tool changer and will stop it at matched tool number. After that, tool changer moves to position by utilizing *tool lever movement* functional block and changes the tool by utilizing *tool clamper*

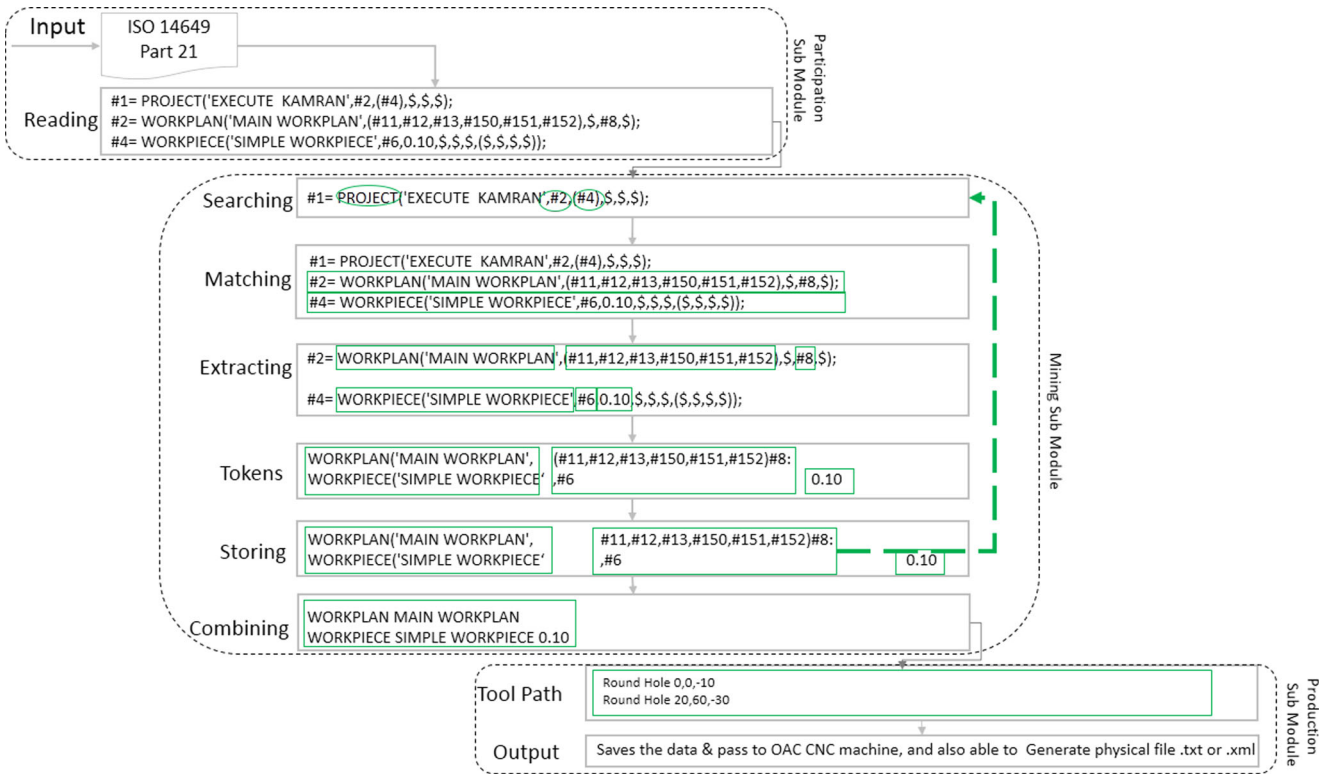

Fig. 4 Algorithm design of STEP-NC interpretation module

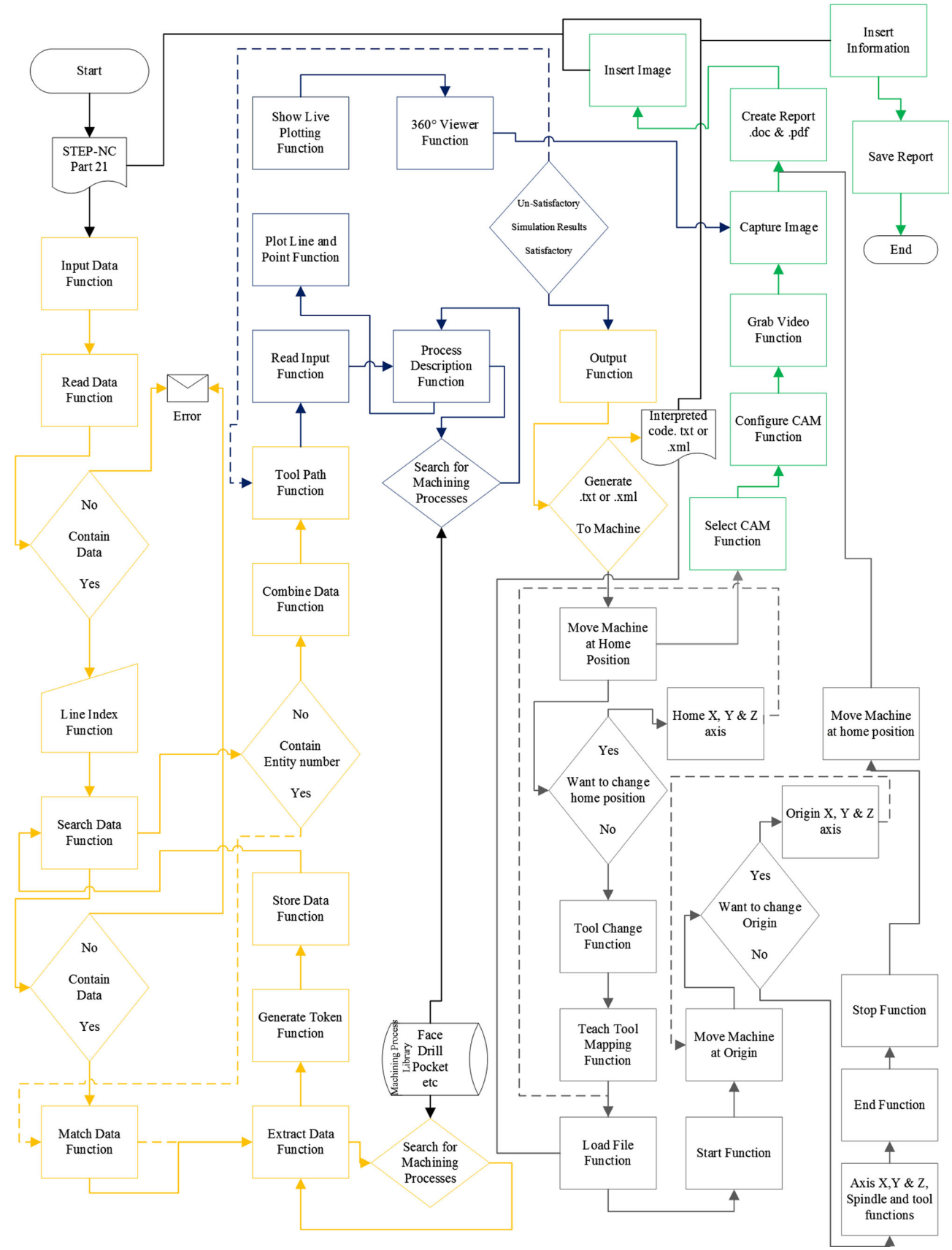

Fig. 5 Algorithm design of the developed system

functional block. In order to avoid accident at the time of tool changing, the algorithm stops the spindle and moves the machine back to home position. After all machining operations are completed, the algorithm ends the script, stops the machine by utilizing *stop* functional block, and also moves the machine back to home position by utilizing *home position* functional block.

After the end of cutting operations, the *capture image* function block is executed to collect and capture the image of input code CAD design, simulation results, and manufactured part in .jpg and formats. After that, the *create word* functional block is executed to create the blank MS word file. After the creation of MS word file, the *insert image* functional block is started to insert the captured images into the created MS word file. At the end of *insert image* function, the *insert information* functional block is activated to, insert input STEP-NC code and interpreted code into the created MS word file. Finally, the *save file* functional block is executed to save the created MS word file in .doc and .pdf format into the system. Figure 5 shows the flow chart of the system algorithm design.

4 GUI of the system

The GUI of the developed system is divided into two portions: *machine motion control* and *interpreter*. The *machine motion control* interface is composed of two tabs: *main* and *settings*. The *main* tab contains *load file* control button that provides a platform for uploading the ISO data interface model file, the uploaded information being displayed in the table format on the *main* tab in shape of separate columns for easy understanding. The *main* tab also contains *spindle on/off* control button for powering on/off spindle motor drive, *coolant on/off* control button to start and stop coolant lubricant, *start* control button to execute the machine, *step* control button to execute machine in step-by-step directions, *stop* control button to stop the machine execution, and *end* control button to end the complete execution. Apart from that, the *main* tab also contains *home* control button that moves the machines at home position, *origin* control button that moves the machine at origin, *stop all motors or X, Y and Z* control buttons that stops X-, Y-, and Z-axis motor drives one by one or all at a time, and *Axis X, Y and Z* indicators that show the real time movement of the CNC machine calculated from rotatory encoders.

In addition, the *main* tab contains functional modules of automatic tool changer settings in “manual function” that contains *drill bit changer*, *drill bit clamper*, *tool change*, *teach tool mapping*, *tool axis jog X, Y, and Z,* and *tool change half* control buttons. The second tab (*settings*) of GUI includes *home x-, y-, and z*-axis control button for machine home position setting, *origin x, y, and z* control buttons for machine origin point setup and *movement settings* control button for settings of maximum and minimum movements of machine tool.

Fig. 6 Snap shots of the developed system GUIs

Fig. 7 Experimental setup

The second portion is of *interpretation* that is composed of six tabs; *main*, *shop floor editor*, *3D simulator*, *monitoring*, *details,* and *settings*. The *main* page contains the *input path* control button from which, the system takes the input file. It also shows the complete contents of the input file with headers data, project name data, work piece data, cutting tool data, and machining process data. The *main* screen shows all the data of header, project name, work piece, cutting tool, and machining processes in different sections, so that user can easily understand the contents of the input code. The *shop floor editor* provides the access to do modifications in the code while machining or off line, where it enables modifications on work piece locations, setup locations, depth of cut, spindle speed, feed rate, processes depth, tolerances, and other information.

The *3D simulation* tab shows the graphical representation of machining operation in 3D graph with process names. The *monitoring* tab shows the image of input code CAD design and live video of machining operations. This tab also provides facilities of drawing editing, quality adjustment, zooming, etc. in live video. This tab has *machining done* control button for image capturing of live video. The *details* tab provides the detailed information regarding setup, work piece, processes, tolerances, locations, depths, etc. in different sections. This tab shows all the data with entity numbers, entity names, values, etc. The last tab is of *settings* that provides access to the input and output setting of the file. Figure 6 shows the snap shots of the developed system GUIs.

5 Experimental study

In order to validate the developed soft-CNC system, an experimental study was conducted. In this study, the manufacturing of ISO 14649 example 1 part and 13 holes drill was carried out by the developed system. The experiment began with the generation of STEP part 21 file by using STEP Tools on line platform for example 1 and GEN-Mill [27] for 13 drill holes. That was followed by the STEP-NC file upload into the developed system. Prior to performing the operation, the setting of the output file was carried out as per requirements from the *settings* tab, upon which completion, the example 1 and 13 holes drill files were executed and the STEP-NC codes translated by *STEP-NC interpretation* module. The translated codes were subsequently verified graphically with the help of *3D simulation* module, before the real machining was carried out by *machine motion control* module and monitored at *monitoring* tab. After successful completion of the machining processes, the document was generated by the *automatic document generation* sub module in .doc and .pdf as shown in Fig. 7.

6 Conclusion

In this study, a novel approach of STEP-based CNC controller was introduced. The scientific contribution of the developed system is the introduction of a new technique for the machine motion control along with STEP-NC data interface model interpretation, graphical verification, automatic

document generation, and live video monitoring. The system also contributes in the utilization of LabVIEW platform for the development of STEP-compliant system. The basic concept of the proposed system was based on the technique developed by Wang et al. and Minhat et al. [20, 21]. However, while these researchers used indirect STEP-NC programming approach, the proposed approach opted for Interpreted STEP-NC programming type. The developed system is composed of *machine motion control*, *STEP-NC interpretation*, *3D simulation*, and *live video monitoring with automatic document generation* modules. The *machine motion control* module was based on the approach taken by Elias et al. While this approach was limited to only open loop three-axis motion control, spindle motion control, and utilized external platform for STEP-NC interpretation [25]. The proposed approach extends the motion control function with automatic tool changer and feedback data (closed loop control system) facilities. All aforementioned facilities enable more flexibility into the CNC system. However, the conversion of old CNC milling machine into a new flexible open CNC systems and automatic document generation has addressed a minute part of sustainable and green manufacturing into the system, respectively.

This developed system enables user friendly environment GUI, where it extracts the complete data from high level codes and shows it in the form of GUI. It shows all the extracted data of setup, work piece, coordinates, locations, depth, tool, tolerance, and others into separate portions to provide easy understanding to user about the content of input file. It translates the input code as per required structure of CNC machine and provides platform of graphical verification of translated code and live monitoring of machining process. This study also shows the validation of the developed system, where it was practically tested and found satisfactory. The development of this kind of systems will be lower in cost as compared to commercial systems. Thus, it can play an important role in the development of small industries and shop floor CNC users.

In future, the system will be updated by providing STEP-NC part 28 interpretation, intelligent and accurate tool path generations, inspection GUI development, intelligent simulation, more processes interpretation, and tool path generation and others. Overall, the main aim of this research is to develop an all-in-one platform for STEP-based CNC system. A similar approach was adopted for ISO 6983 data interface model [28, 29].

Acknowledgments This work was supported by the Malaysian Government under Science fund (MOSTI) Vote No S02, DRB-HICOM University of Automotive Malaysia (DHU), Malaysian International Scholarship (MIS) under Ministry of Education (MOE) Malaysia and University Tun Hussein Onn Malaysia (UTHM). The authors would like to thank Dr. David Loffredo Vice President of STEP Tools Inc for his technical support in this study.

References

1. Suh SH, Cheon SU (2002) A framework for an intelligent CNC and data model. Int J Adv Manuf Technol 19:727–735
2. Suh SH, Cho JH, Hong HD (2002) On the architecture of intelligent STEP-compliant CNC. Int J Comput Integr Manufac 15:168–177
3. TC184, I. SC4 (1991) Product data representation and exchange-part 1: overview and fundamental principles. ISO/DIS
4. Dunn M (1992) Industrial automation systems and integration—product data representation and exchange - part 48. Integration generic resources form features ISO/WD
5. Newman S, Nassehi A, Xu X, Rosso R, Wang L, Yusof Y, Ali L, Liu R, Zheng L, Kumar S (2008) Strategic advantages of interoperability for global manufacturing using CNC technology. Robot Comput Integr Manuf 24:699–708
6. Xu X, Newman S (2006) Making CNC machine tools more open, interoperable and intelligent - a review of the technologies. Compu Ind 57:141–152
7. Rauch M, Laguionie R, Hascoet JY, Suh SH (2012) An advanced STEP-NC controller for intelligent machining processes. Robot Comput Integr Manuf 28:375–384
8. Hamilton K, Hascoet JY, Rauch M (2014) Implementing STEP-NC: exploring possibilities for the future of advanced manufacturing. In: modern mechanical engineering. Springer, pp 199–239
9. Mori M, Yamazaki K, Fujishima M, Liu J, Furukawa N (2001) A study on development of an open servo system for intelligent control of a CNC machine tool. CIRP Ann Manuf Technol 50:247–250
10. Asato O, Kato E, Inamasu R, Porto A (2002) Analysis of open CNC architecture for machine tools. J Braz Soc Mech Sci 24:208–212
11. Zheng J, Zhao W, Li Z (2005) Open EDM CNC system based on RT Linux. Comput Integr Manuf Syst 11:1179
12. Park S, Kim SH, Cho H (2006) Kernel software for efficiently building, re-configuring, and distributing an open CNC controller. Int J Adv Manuf Technol 27:788–796
13. Ma XB, Han ZY, Wang YZ, Fu HY (2007) Development of a PC-based open architecture software-CNC system. Chin J Aeronaut 20:272–281
14. Suh SH, Lee BE, Chung DH, Cheon S (2003) Architecture and implementation of a shop-floor programming system for STEP-compliant CNC. Comput -Aided Des 35:1069–1083
15. Xu X (2006) Realization of STEP-NC enabled machining. Robot Comput Integr Manuf 22:144–153
16. Weck M, Wolf J, Kiritsis D (2001) STEP-NC—the STEP compliant NC programming interface. Int Intell Manuf Syst Forum
17. Wolf J (2001) STEP-NC - integrating shop floor into industrial data flow for the enabling of intelligent near to process functions, Korea-Germany Workshop on STEP-NC, 23
18. Storr A, Pritschow G, Heusinger S, Azotov A (2002) Working-step planning for turning with STEP-NC: planning methods for user support. IWF Zeitschrift fur Wirtschaftlichen Fabrikbetrieb 97:390
19. Erdos G, Xirouchakis P (2003) STEP-NC data model developement for wire-EDM manufacturing. IFAC
20. Wang H, Xu X, Des Tedford J (2007) An adaptable CNC system based on STEP-NC and function blocks. Int J Produc Res 45:3809–3829
21. Minhat M, Vyatkin V, Xu X, Wong S, Al Bayaa Z (2009) A novel open CNC architecture based on STEP-NC data model and IEC 61499 function blocks. Robot Comput Integr Manuf 25:560–569
22. Weidong Y, Zhanbiao G (2010) An open CNC controller based on LabVIEW software. Comput Appl Syst Model (ICCASM) IEEE North Univ Chin 4:476–479

23. Zhanbiao G (2010) An open CNC controller based on LabVIEW software. In: Computer application and system modeling (ICCASM) IEEE North University of China
24. Da Rocha P, Diogne R, de Silva e Souza ME, De Lima Tostes (2010) Prototype CNC machine design. In: Industry applications (INDUSCON). IEEE, Sao Paulo, pp 1–5
25. Elias D, Yusof Y, Minhat M (2013) CNC machine system via STEP-NC data model and LabVIEW platform for milling operation. In: Open systems (ICOS). IEEE, pp 27–31
26. Elias D, Yusof Y, Minhat M (2014) An open STEP-NC controller via labview platform. Appl Mech Mater 660:873–877
27. Yusof Y, Kassim ND, Zamri Tan NZ (2011) The development of a new STEP-NC code generator (GEN-MILL). Int J Comput Integr Manuf 24:126–134
28. Yusof Y, Latif K (2015) A novel ISO 6983 interpreter for open architecture CNC systems. In: The international journal of advanced manufacturing technology, pp 1–10
29. Yusof Y, Latif K (2015) New technique for the interpretation of ISO 14649 and 6983 based on open CNC technology. In: International journal of computer integrated manufacturing, pp 1–13