Robotics and Computer–Integrated Manufacturing 51 (2018) 25–36

Contents lists available at ScienceDirect

Robotics and Computer–Integrated Manufacturing

journal homepage: www.elsevier.com/locate/rcim

Full length Article

A novel command generation paradigm for production machine systems

Ulas Yaman*, Melik Dolen

Department of Mechanical Engineering, Middle East Technical University, Ankara 06800, Turkey

ARTICLE INFO

Keywords:
Command generation
Machine control languages
CNC machine tools
3D Printers
Curve offsetting
Vector operations
Python scripting language

ABSTRACT

This paper presents a new command generation technique titled VEPRO for computer controlled production machinery. In this method, the tool trajectory is described by a high-level scripting language that enables parametric representations of complex work-piece geometries. The interpreted script is then employed to generate the interpolation data required to compute a tool trajectory that is subjected to a number of kinematic constraints. A real-time interpolator is employed to provide the position commands required by each motion controller in a synchronous fashion. As a proof of concept, the proposed method is emulated on a PC using Python scripting language. The performance of the paradigm is rigorously assessed via two test cases involving different manufacturing techniques (e.g. pocket milling and 3D printing). Through the experimental results, the paper illustrates that the technique, which lends itself for real-time hardware implementation, exhibits satisfactory performance for all intensive purposes and that the method is technically feasible for a wide spectrum of manufacturing applications and systems.

1. Introduction

Command generators (CGs) serve as key components for computer-controlled production machines, industrial robots, advanced automation systems and more. They simply calculate the reference positions of tool/end effector on predefined trajectory so that the corresponding information (along with the higher-order time derivatives of the reference if deemed necessary by the control algorithm) are synchronously fed to the digital motion controller of a particular machinery at each and every sampling period.

In manufacturing industry, the command trajectory (along with the accompanying auxiliary functions) is defined via machine control languages (MCLs) conforming to various international- or industrial "de-facto" standards. These languages include numerical control (NC) language (ISO 6983, EIA 274D) [1], BCL (EIA 494C) [2], STEP-NC (ISO 10303-238) [3], and DMIS (ISO 22093) [4]. The automatically programmed tools (APT) [5] and robot programming languages (e.g. AML, HELP, Karel, RAIL, VAL 3, etc.) [6] can be regarded as high-level programming languages to produce operational instructions for the corresponding applications. A concise discussion of the afore-mentioned MCLs along with the issues for selecting a proper standard for a specific application is further discussed by [7]. As of today, the NC language (a.k.a. "G-code") as described by ISO 6983 is widely adapted in manufacturing industry. The (machine-specific) low-level language essentially defines the tool trajectory in terms of line-, arc-, and complex curve segments (parabola, helix, spline, etc.). The versatility of the NC language is often-times enhanced by the functions (such as advanced code editing, syntax highlighting, macro definitions, machining simulation/visualization, etc.) associated with graphical-user interfaces (GUIs). Furthermore, some CNC unit manufacturers extend the NC language via high-level language constructs (i.e. conditional branching, loop controls, arithmetic operations, etc.) to improve the capabilities of the NC programs (e.g. Sinumerik high-level language). Despite enhancements through CAM-like features, the underlying CG scheme remains unchanged and the accompanying problems are somewhat inherited [7,8].

To overcome the problems for the existing CG scheme within the context of ISO 6983, a new standard called STEP-NC (ISO 10303-238) has been established to create a new manufacturing infrastructure along with next generation of "intelligent" CNC machine tools. The underlying data model for STEP-NC incorporates the complete information on how to manufacture a workpiece including its geometry, tools/fixtures involved, work plans, tool paths, CNC model/type, and more. Due to its comprehensive content, the CNC machine tools employing STEP-NC are still in their early development stages after (more than) a decade of intense research [9–14] and commercial STEP-NC machine tools are very rare. To maintain applicability, the current research efforts mostly employ STEP-NC translators to generate NC codes to make good use of the industry de-facto standard CNC units. On the other hand, the SPAIM (STEP-NC Platform for Advanced Intelligent Manufacturing) architecture [9,10] developed within the context of the FoFdation project

* Corresponding author.
E-mail addresses: uyaman@metu.edu.tr (U. Yaman), dolen@metu.edu.tr (M. Dolen).

https://doi.org/10.1016/j.rcim.2017.11.016
Received 14 March 2017; Received in revised form 15 November 2017; Accepted 16 November 2017

[15] is one of the exceptions where a complete STEP-NC enabled CNC system was developed by making good use of LinuxCNC [16] (OpenNC) platform.

An alternative solution is proposed in this paper. In this approach, a universal code called VEPRO (acronym for **VE**ctor **PRO**cessor) is utilized as the central piece in CG scheme.The code, which includes sophisticated functions as deemed by production machinery, is a high-level language enabling parametric representation of tool trajectories. Despite some minor similarities with the APT language and the STEP-NC; the technique, which embraces to bottom-up hierarchical integration, differs significantly from its counterparts in terms of implementation, structure, information model, and functionality. The novel features of the proposed language are as follows:

- VEPRO is based on (open-source) Python language which is easy to master whereas the (obsolete) APT language constitutes 576 vocabulary words [17] while the STEP-NC standard alone covers more than 1500 pages [3].
  - VEPRO is a dynamic language which is extendable for new applications. New libraries, functions, and data structures could be incorporated as the need arises in the future.
  - Note that for STEP-NC, the work is still underway by various ISO Committees. The extension of the standard to new manufacturing processes (such as contour cutting, Wire EDM, Sink EDM) is currently under development.
- VEPRO enables the development of parametric programs (or templates). Such programs could be conveniently modified to characterize a family of different geometries for workpieces. Given the critical geometric entities, a process engineer can manually develop/edit the VEPRO code suitable for a particular production machinery. VEPRO programming is expected to become much more versatile when fully supported by specialized GUIs and utility programs in the near future.
- VEPRO embraces “bottom-up” approach where low-level functional units (e.g. interpolation, low-level supervision, feed-modulation, data management) are built, tested, and organized into more complex entities (e.g. application objects, manufacturing strategies/plans, intelligent agents, etc.) up in the hierarchy to handle sophisticated production tasks.
- VEPRO language currently provides unidirectional information flow in CAD/CAM/CNC chain. However, VEPRO could evolve to accommodate bidirectional information flow in the future.
- Unlike APT (or STEP-NC), VEPRO does not directly employ abstract geometric features/entities. Instead, the basic data structures of VEPRO are jagged arrays that are handy while devising/editing/debugging part programs.
- Just like STEP-NC, VEPRO programs hold the prospect of offering full portability on a family of production machinery provided that the VEPRO interpolator unit is realized and standardized in time.
- Unlike the APT processor, the VEPRO could be simply implemented on small-form factor computers running the Python interpreter.
  - STEP-NC processor code has yet to be developed/(fully) tested for the newer generations of microprocessors/major computing platforms and is likely to be proprietary code.
- VEPRO can co-exist with the software tools/systems employed in the current CNC infra-structure.
  - VEPRO could be incorporated into the existing CNC units and could enable its users to work with VEPRO- and NC codes side by side.
  - To increase the efficiency of the programming phase, the automatic code generation could be implemented as a part of CAM/CAPP system. Since on functional level, VEPRO is directly compatible with most CAM software tools, VEPRO code could be automatically generated with the utilization of proper add-on modules.

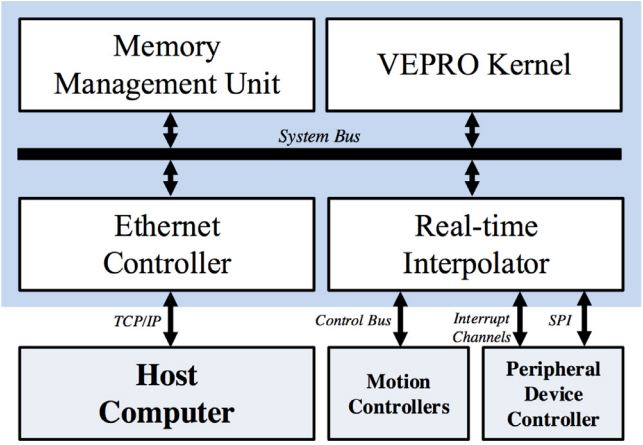

Fig. 1. Architecture of Vector Processor (VEPRO).

  - As an interim solution, CLDATA could be facilitated to generate the required code without augmenting the existing manufacturing infrastructure.

Table 1 describes the properties of different platforms. Please note that STEP-NC standard itself does not specify how the data from the models are utilized to generate tool paths (or interpolation techniques for that matter) [18,19]. On the other hand, VEPRO is an advanced interpolation scheme which could complement the operation of conventional- and STEP-NC compliant manufacturing systems. Consequently, the objective of this paper is to assess the potential for devising a VEPRO based CGs for a wide spectrum of computer controlled production machinery.

The organization of this paper is as follows: After this introduction, the following section elaborates the VEPRO architecture while the next one discusses the corresponding high-level language including some of the critical functions developed for common manufacturing tasks. Section 4 evaluates the presented technique experimentally on two test cases involving machining and 3D printing. Finally, some key results about this study are given in the last section.

2. Architecture

VEPRO is a motion-command generator designed specifically for CNC machine tools, industrial manipulators, and advanced machinery used in manufacturing industry. As shown in Fig. 1, the system is composed of four major units: (i) VEPRO Kernel; (ii) Real-time Interpolator; (iii) Memory Management Unit; (iv) Ethernet Controller. Despite the fact that the system-on-a-chip (SoC) solution could be realized on a FPGA as suggested by [20], the paper does not dwell on hardware implementation details and primarily focusses on the architecture on functional level. A detailed discussion of the SoC implementation of the system including multi-processor version, is given in [20]. It is interesting to note that [21] proposes a CG architecture incorporating two abstraction layers where the assembly-like language (called BNCL) is utilized to represent the trajectory on one layer (BVM) while the machine hardware is abstracted by another (BVH). Despite some similarities in concept, the presented approach here differs significantly from the above-mentioned architecture in terms of functional organization, language, data abstraction, information flow, and most importantly implementation.

The brief descriptions of these units follow:

- **Memory management unit (MMU)** manages the data traffic among the VEPRO units and the memory devices (i.e. SRAM, SDRAM, E2PROM, flash/solid-state drives, SD card etc.). For the sake of simplicity, the MMU employs a (byte-indexed) 32-bit addressable linear memory model. A portion of the memory (called hardware page) is especially dedicated to file/control registers where the active data