Journal of Manufacturing Systems xxx (2016) xxx–xxx

Contents lists available at ScienceDirect

Journal of Manufacturing Systems

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

Voxel model surface offsetting for computer-aided manufacturing using virtualized high-performance computing

Roby Lynn[a,*], Didier Contis[b], Mohammad Hossain[c], Nuodi Huang[d], Tommy Tucker[e], Thomas Kurfess[a]

[a] Georgia Institute of Technology, George W. Woodruff School of Mechanical Engineering, Atlanta, GA, USA
[b] Georgia Institute of Technology, College of Engineering, Atlanta, GA, USA
[c] Georgia Institute of Technology, Department of Computer Science, Atlanta, GA, USA
[d] Shanghai Jiao Tong University, School of Mechanical Engineering, Shanghai, China
[e] Tucker Innovations, Inc., Charlotte, NC, USA

ARTICLE INFO

Article history:
Received 23 September 2016
Received in revised form
12 November 2016
Accepted 16 December 2016
Available online xxx

Keywords:
Cloud manufacturing
Distributed manufacturing
Computer-aided manufacturing
Computer numerical control
Virtualization
High-performance computing
Voxel
GPGPU
CUDA

ABSTRACT

Curve and surface offsetting is a common operation performed on solid models when planning toolpaths for a machining operation. This operation is usually done in a computer-aided manufacturing (CAM) software package to define the path along which the center of a cutting tool will follow to create a given feature. The CAM software then translates the toolpath created from the offset into G-Code, which is the standard programming language of CNC machine tools. The toolpath planning process can be computationally intensive; thus, a powerful workstation is required to run the CAM software effectively. These standalone workstations can be inconvenient due to their cost and size. Many organizations have been turning to virtualization as an alternative to multiple standalone workstations; virtualization allows for many users to access desktop environments that are hosted from a single remote server. This has the benefit of isolating the user from both OS and hardware requirements for certain software, and also allows them to run the applications they need from anywhere. This research explores the emerging area of virtualized general purpose computation on graphics processing units (GPGPU); this technique is used to support the development of a voxelized CAM package that allows for rapid toolpath generation for complex parts. The surface offset performance is benchmarked on various local and virtualized platforms to evaluate the losses from virtualization. Results indicate a minor loss of performance between virtualized and local GPUs, which is to be expected due to the abstraction of hardware from a virtual machine. Additionally, the development of a GPU-sharing implementation using a server operating system is described and analyzed; results indicate that, as compared to single virtual machines, the GPU-sharing approach demonstrates higher computational efficiency with the addition of compute load to the GPU.

1. Introduction

Machining is a decades-old manufacturing process that is in wide use today because it enables manufacturers to create parts with complex geometry and tight tolerances. Originally, machining was performed using manual equipment where the motion of the cutting tool was controlled with handwheels. Today, modern machining operations are performed using computer-numerical control (CNC) machine tools, where the tool motion is controlled by servomotors. The machine tool is programmed using G-Code, which is a collection of movement commands for the machine tool that describe the sequence of tool motions needed to create a part. The G-Code for the part can be created either manually or through the use of CAM software. For complex parts that require many machine axes, such as 5-axis parts, CAM software is essential to the rapid creation of accurate toolpaths.

CAM software, while incredibly useful, can also be difficult to use properly. The CAM programmer must have machining experience to be able to determine appropriate machining parameters for order-of-operations; additionally, the operation of the software itself requires specialized training [1]. Many manufacturers and other machine tool professionals have expressed the need for

* Corresponding author.
E-mail addresses: roby.lynn@gatech.edu (R. Lynn), didier.contis@coe.gatech.edu (D. Contis), mhossain7@gatech.edu (M. Hossain), huangnuodi@126.com (N. Huang), tommy@tuckerinnovations.com (T. Tucker), kurfess@gatech.edu (T. Kurfess).

http://dx.doi.org/10.1016/j.jmsy.2016.12.005
0278-6125/

easier-to-use CAM software [2]. Additionally, the machining simulations provided by current CAM software are not able to accurately represent gouging and toolmarks that are left behind once machining is complete. A better approach is needed that is both easier to use and provides more realistic simulation of volume removal during the machining process to provide better feedback to the CAM programmer.

Typical CAM software relies on analytical models to describe part geometry, such as a boundary representation (B-rep). While these models are compact and easy to store in memory, their complexity is limited by the precision of the computer that is used to process them. Additionally, limitations exist in the modification of the part surfaces when simulating material removal during the machining process [3]. To overcome these limitations, this work proposes the use of a voxel-based part model where the entire part is discretized into small cubes. Similar to a two-dimensional image, which is made up of small pixels, the voxel model is made up of small cubes called voxels. The structure of the voxelized part model, known as a Hybrid Dynamic Tree (HDT), allows for simplified computations of Boolean volume subtraction during the simulation of the machining process, which can allow for more accurate visualization and feedback to the CAM operator [4,5]. The voxel model, however, requires a different strategy than an analytical model to process efficiently. Because the voxel model is effectively a large array, different operations on the array can be performed simultaneously to accelerate the speed at which the model is processed. These operations can be performed on a parallel computing platform, such as a graphics processing unit (GPU). The GPU implementation of the HDT has been shown to greatly outperform a comparable implementation that relied on a single CPU thread [6].

While the use of the voxel model allows for distinct advantages over traditional analytical models, the speed at which the model can be processed is dependent upon the performance of the parallel computing platform that is employed. For this implementation, NVIDIA graphics cards with compute unified device architecture (CUDA) functionality were employed. CUDA is an application programming interface (API) that allows for access to the GPU for computational instead of graphics purposes. Many modern computers are equipped with discrete GPUs, which allows for use of the software without additional hardware. However, many mobile devices, including laptops and tablets, do not have discrete GPUs that can be used for computation. For users without access to a GPU compute device, virtualization can be employed to allow them to access a computer with GPGPU capability.

Virtualization is the implementation of a simulated desktop on a computer or server. This is accomplished by abstracting the physical hardware of the machine. In the most basic sense, virtualization can be used to run an instance of a guest operating system (OS) on top of an already running host OS; for example, a user could start a Linux session using virtualization software inside Windows to allow them to simulate a Linux computer even though the native OS is Windows. Virtualization is frequently used to host multiple desktops on a single server; this allows for many users to access the same hardware and each have an individual desktop. Powerful hardware can be installed in the server, and users can access it from any internet connected device. Thus, the clients do not need physical access to a GPU compute device, as the hardware can be virtualized from the remote server. While the virtualization of GPUs for graphics rendering is commonplace, virtualization for GPGPU is less widespread. Commercial virtualization solutions support this functionality to a limited extent; however, there is a deficiency of solutions that allow for multiple users to share a single GPU for compute tasks. The present work explores a time-sharing approach to GPU virtualization that allows for many simultaneous users to perform toolpath planning on a single GPU.

The remainder of this paper is organized as follows: first, background information in presented on the current states of both the virtualization of engineering software and the use of GPU-acceleration in engineering computation; second, the state of virtualized GPUs for computation is discussed. Next, a user-friendly voxelized CAM package called SculptPrint is introduced. SculptPrint relies on the GPU to process a voxel model and allows for very accurate simulation of the machining process. Further technical details about SculptPrint and the voxel model are discussed. Next, the experimental setup to evaluate the performance of SculptPrint on a virtualized platform is discussed and results are presented. Discussions and conclusions evaluate the practicality, advantages and limitations of the virtualized implementation.

2. Related Work

Cloud manufacturing is the idea of leveraging cloud computing resources to facilitate manufacturing; this can be done through distributed computing, virtualization, etc. [7]. Multiple researchers have shown that virtualization is an important tool for realizing higher security and more widespread deployment of large applications in the manufacturing environment [8,9]. Virtualization allows users to remotely access high-end HPC (high performance computing) hardware from any device and leverage the power of that hardware without being physically close to it [10]. This is an example of infrastructure-as-a-service (IaaS), where the actual computing hardware is provided to users for them to run computations on. Typical engineering software, such as computer-aided design (CAD), computer-aided engineering (CAE) and CAM packages require hardware acceleration for optimal performance. This usually necessitates a powerful CPU, large amounts of RAM and a discrete GPU. The GPU is the most critical component for realizing high-performance parallel computation on a single machine [11]. To remove these hardware requirements from the client machine, engineering software can be virtualized and presented to the user on a virtual desktop.

2.1. Virtualization of engineering software

Hardware-accelerated virtual desktops have been used for CAD/CAM software virtualization in large organizations previously [12–15]. Recent improvements in display protocols, such as Citrix ICA, Microsoft RemoteFX and Teradici PCoIP, combined with hypervisor improvements and the emergence of graphics virtualization technologies, have supported the adoption of VDI (virtual desktop infrastructure) for CAD/CAM workload [16]. Virtualization allows for members of an organization to utilize hardware-accelerated software from thin clients or personal machines without powerful hardware; it also removes operating system restrictions imposed by certain software, as the client only duplicates the display for the user and the software itself is run on the virtual machine. Application security and ease of troubleshooting can be increased through the use of a private cloud that has a controlled user base. The implementation of virtualized desktops requires a hypervisor, for example Microsoft Hyper-V, Citrix XenServer or VMWare vSphere. The hypervisor is responsible for dividing the physical resources of the server amongst the VM sessions. The hypervisor requires computational resources to run, and thus performance of virtual machines is not identical to that of standalone machines [17]. While this can be a problem, more powerful server hardware can be selected to mitigate performance losses. Additionally, some virtualization solutions, such as Citrix XenServer, grant VMs direct access to hardware resources [18], which bypasses parts of the abstraction layer that causes slowdowns in traditional virtualization. When applied to GPUs, this is known as PCI-passthrough: the PCI device is

Please cite this article in press as: Lynn R, et al. Voxel model surface offsetting for computer-aided manufacturing using virtualized high-performance computing. J Manuf Syst (2016), http://dx.doi.org/10.1016/j.jmsy.2016.12.005