`

A Three-tier Redundant Architecture for Safe and Reliable Cloud-based CNC over Public Internet Networks

C.E. Okwudire[a, *], X. Lu[a], G. Kumaravelu[a], and H. Madhyastha[b]

[a]*Smart and Sustainable Automation Research Laboratory, Department of Mechanical Engineering, University of Michigan, Ann Arbor, USA*
[b]*Department of Computer Science and Engineering, University of Michigan, Ann Arbor, USA*

Keywords:
Cloud manufacturing
Computer Numerical Control (CNC)
Control as a Service
Cyber-physical system

ABSTRACT

Cloud-based CNC is an emerging paradigm of Industry 4.0 where computer numerical control (CNC) functionalities are moved to the cloud and provided to manufacturing machines as a service. Among many benefits, C-CNC allows manufacturing machines to leverage advanced control algorithms running on cloud computers to boost their performance at low cost, without need for major hardware upgrades. However, a fundamental challenge of C-CNC is how to guarantee safety and reliability of machine control given variable Internet quality of service, especially on public Internet networks. We propose a three-tier redundant architecture to address this challenge. We then prototype tier one of the architecture on a 3D printer successfully controlled via C-CNC over public Internet connections, and discuss follow-on research opportunities.

1. Introduction

As part of Industry 4.0, manufacturing resources are increasingly being provisioned via cloud manufacturing as services from the cloud, for all phases of the product development lifecycle [1]-[6]. Consider the traditional computer integrated manufacturing workflow (Fig. 1(a)), consisting of computer-aided design, manufacturing and process planning (CAD, CAM, CAPP) together with computer numerical control (CNC) all running on local computer hardware [4]. In Industry 4.0, this workflow will be transitioned to a cloud ecosystem where its various elements run on virtualized (cloud) computers (see Fig. 1(b); note that C- denotes cloud-based). Users will be able to connect to this ecosystem from anywhere in the world to design and manufacture parts on-demand using a variety of manufacturing machines (MMs) connected to the ecosystem. To this end, there already exist examples of C-CAD (e.g., OnShape [7]), C-CAM (e.g., VisualCAMc [8]) and C-CAPP (e.g., MachiningCloud [9]); and CNC with cloud connectivity is becoming commonplace, thus opening up opportunities for a transition to C-CNC [5].

The potential benefits of C-CNC have been highlighted in recent articles [10]-[14]:

1) With C-CNC, a MM can be controlled using advanced algorithms that may not be executable on a local controller with limited computational resources; C-CNC thus holds the potential to significantly improve the performance of MMs without need for upgrades to powerful and costly computational hardware.
2) C-CNC allows a user to opt for only the CNC functionalities they need, when they need it, thus slashing costs.
3) With C-CNC, upgrading the controller of a machine becomes much easier, involving little or no hardware changes. This reduces the problem of frequent obsolescence of MMs due to outdated control hardware.
4) Due to the connected infrastructure provided by C-CNC, data analytics can be used to improve the performance of control algorithms (e.g., fine-tune controller parameters) based on feedback from several machines running the algorithms.
5) With C-CNC, the often-frustrating barriers between CAD, CAPP, CAM and CNC in the traditional computer integrated manufacturing paradigm can be dissolved, allowing for seamless information transfer among the cloud-based versions of the various components. For instance, Gcode, which has a poor representation of design intent, can be eliminated altogether [4].

There are, however, two major objections to C-CNC. The first is cybersecurity. This is a valid concern raised against virtually all cloud-based services, and often slows down adoption of cloud solutions [6]. However, companies (like Amazon) which provide platforms for cloud-based applications are able to guarantee close to 100% security [1]. Hence, security-critical applications (like banking) are increasingly adopting cloud solutions. Techniques, like those used for cloud banking, can be used in cloud manufacturing [5][6]. Hence, one can argue that, though critical, cybersecurity is not the key inhibitor to C-CNC from a practical standpoint.

* *Corresponding author.* Tel.: 1-734-647-1531.
E-mail address: okwudire@umich.edu.

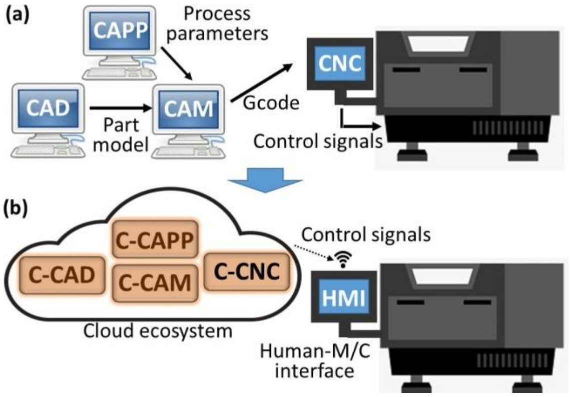

Fig. 1 - (a) Traditional computer integrated manufacturing workflow; (b) Cloud manufacturing automation paradigm for Industry 4.0.

The second concern is that sending real-time control signals over the Internet introduces safety and reliability risks due to time delays, dropped packets, and other Internet quality of service (QoS) issues [10]-[14]. The decades-old field of Networked Control Systems has studied this issue in the context of feedback controllers, but the resultant solutions are based on Lyapunov stability criteria, which are conservative; i.e., they overly sacrifice performance to guarantee closed loop stability [15]. Recently, the nascent field of control as a service (CaaS) [10]-[14], [16]-[20] has advocated the use of edge/fog computing [19] or private cloud infrastructure [20] to mitigate QoS issues. The idea is to use specialized solutions like software defined networking (SDN) [21] and time sensitive networking (TSN) [22] to improve Internet QoS. However, these specialized solutions are likely to significantly limit the reach and increase the cost of C-CNC, especially for small and medium size enterprises (SMEs), which are poised to benefit the most from cloud solutions [23]. It is, therefore, desirable to have a solution that makes C-CNC broadly available over public/shared Internet connections, much like C-CAD, C-CAM and C-CAPP. One way of doing this is to transition only feedforward (FF) control functionalities of CNC (like trajectory generation and feedrate optimization) to the cloud, while leaving the time and stability-critical feedback (FB) loops in the local CNC [10]-[14].

In the context of FF-focused C-CNC over public Internet networks, Schlechtendahl et al. [11] observed severe QoS issues (like long delays and data losses) in transmitting data. Sang and Xu [12] proposed a FF-focused C-CNC architecture for switching to a local CNC to safely stop a MM if QoS deteriorates unacceptably. However, it is more desirable to guarantee safety without unduly stopping the MM, to avoid excessive loss of productivity. As its original contributions, in Section 2, this paper makes a case for adding some FB to FF-focused C-CNC and proposes a three-tier redundant architecture to guarantee safe and reliable C-CNC, without stopping the MM. A prototype of tier one of the proposed architecture on a 3D printer is then presented in Section 3. Further research opportunities emanating from the proposed architecture are discussed in Section 4, followed by conclusions and future work.

2. C-CNC with feedback and redundancy

2.1. Need for some Feedback in FF-focused C-CNC

Recent work, e.g., [10]-[14], has made a strong case for C-CNC where only FF control functionalities are hosted in the cloud while FB loops are retained in a local CNC. By FF control, we mean all techniques that rely entirely on a priori known information to generate MM control commands (including setpoints). Examples include multi-axis B-spline interpolation, trajectory generation, feedrate optimization and model-based error compensation.

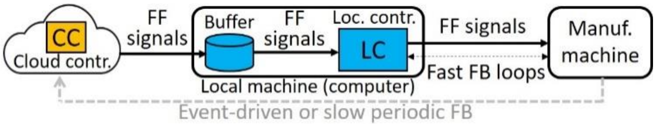

Fig. 2 - Elements of FF-focused C-CNC (outlined in black color) combined with event-driven or slow periodic FB updates, as advocated in this paper (outlined in grey color).

With focus on only FF control (outlined in black color in Fig. 2), the QoS issues related to C-CNC can be resolved by: (1) transmitting control signals from the cloud controller (CC) to the MM using an Internet protocol that ensures the accuracy of the transmitted data and (2) provisioning a large enough buffer on the receiving local machine (i.e., local computer) to absorb worst-case transmission delays. The FF control signals are then fetched from the buffer and delivered in real-time to the MM via a local controller (LC). Note that the LC is also responsible for handling fast real-time FB control loops, if present, e.g., servo position control.

However, there are usually unanticipated circumstances that may require rapid changes to the FF control signals calculated by a CC. For example, a MM operator may issue a feed (motion speed) override, or a safety sensor may be triggered, requiring sudden changes in the FF signals of the MM. Event-driven FB is needed to allow FF-focused C-CNC to account for such circumstances, e.g., to re-plan trajectories accordingly. Moreover, a model-based CC in FF-focused C-CNC may need some FB to help calibrate its parameters periodically. An example of this is the virtually assisted milling force adaptive controller proposed by Altintas and Aslan in [24]. A computer model (digital twin) of a milling process was run on a local computer and used to calculate expected force levels when milling a part. In parallel, measurements (or observations) of forces were acquired from the milling machine. The simulated forces were used in real-time to calibrate the force thresholds of the adaptive controller thus enhancing its accuracy. Simultaneously, the forces obtained from the milling machine were used to calibrate the parameters of the digital twin to improve its accuracy. With C-CNC, the digital twin would be run in the cloud as part of a CC, but it would need the periodic FB from the milling machine and the ability for the CC to respond fast enough to the influence of FB on the FF commands.

Therefore, in this paper, we advocate for C-CNC (as shown in Fig. 2) to: (1) include event-driven and/or slow periodic FB updates (e.g., in the order of 1 Hz); and (2) enable the machine to benefit quickly from changes to FF control signals due to the FB updates (e.g., by having a small buffer, which makes C-CNC more susceptible to variable transmission delays.

2.2. Three-tier Redundant Architecture for C-CNC

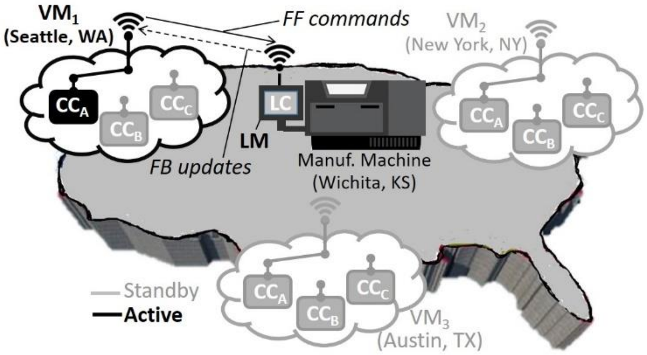

Fig. 3 - Schematic of proposed three-tier redundant architecture exemplified using the United States.

Internet QoS can deteriorate at any time, especially when signals are transmitted over public WANs, as highlighted by Schlechtendahl et al. [11]. It is critical for a MM controlled by C-CNC to be safe irrespective of QoS; i.e., it should never lose control and should be able to respond quickly to emergencies (like feed override). Moreover, it should be reliable, meaning that one can count on its availability and acceptable performance even when QoS degrades.

To guarantee the safety and reliability of C-CNC, we propose the redundant architecture exemplified in Fig. 3 using the U.S.; it is loosely borrowed from concepts in cloud computing. It consists of a set of virtual machines (i.e., cloud computers) – VM$_1$, VM$_2$, VM$_3$, etc. – as well as a local machine (LM) tethered to a MM. As is typical in cloud platforms, the VMs are assumed to be geographically distributed and heterogeneous with respect to pricing. It is assumed that, for a given control application (e.g., milling force adaptive control [24]), various grades of cloud controllers, CC$_A$, CC$_B$, CC$_C$, etc., can be configured to run on each VM with decreasing communication overhead and performance. The active CC (e.g., CC$_A$ of VM$_1$ in Fig. 3) pre-computes FF control commands over a given time horizon and sends them via the Internet to be buffered in the LM to accommodate transmission delays. However, the time horizon for buffering is limited by the memory of the LM and/or constraints on the response time of the CC to FB updates (e.g., every few seconds). The standby LC, CCs and VMs (see Fig. 3) provide three tiers of redundancy in the C-CNC architecture, as described below.

2.2.1 Tier 1: Safety and base level reliability using backup LC

In the proposed architecture (Fig. 3), C-CNC is assumed to have a basic LC that: (i) runs the fast real-time FB loops (e.g., servo position control) which are not migrated to the cloud; and (ii) can handle safety emergencies or maintain FF control of the MM by switching from the currently active CC to the LC. For example, if a feed override is issued or a proximity sensor detects an impending collision, there is no guarantee that the CC can act fast enough to re-plan and deliver new FF control signals, because of uncertain delays that occur in transmitting signals over the Internet [11]. In such circumstances, we propose that the LC takes over smoothly and quickly from the active LC to re-plan the FF control signals. Moreover, once QoS improves, a quick and smooth handover from the LC back to the active CC is proposed in our architecture, to maximize the benefits of C-CNC.

2.2.2 Tier 2: Enhanced reliability via backup cloud controllers

We assume that in C-CNC the CC is much more superior in performance to the associated LC, to justify the adoption of C-CNC over local CNC. Therefore, it is not desirable to have no option other than to switch to the LC whenever QoS deteriorates, otherwise the overall performance of C-CNC will be severely diminished. To address this, Tier 2 borrows an idea from adaptive bitrate (video) streaming [25] where video quality is slightly downgraded to ease communication overhead when QoS deteriorates. Similarly, in our proposed C-CNC architecture, the CC is assumed to have several performance grades (CC$_A$, CC$_B$, CC$_C$, etc.) that are all better than the LC. If QoS deteriorates, C-CNC has the option of switching to the lower grades of CC while leaving the LC as the last resort, thereby minimizing loss of performance. An example of a lower-grade CC would be one that reduces the number of data points transmitted per second, at the cost of reduced speed or precision of the associated MM.

2.2.3 Tier 3: Enhanced reliability via backup cloud computers

In cloud computing applications, it is common to enhance reliability by having a number of standby servers ready to run a given cloud application [26]. A standby server is engaged if QoS deteriorates unacceptably along an active communication path. Tier 3 adopts this idea as an alternative (or complement) to Tier 2. CCs are installed on geographically dispersed backup cloud computers (e.g., VM_1, VM_2, VM_3 in Fig.3), often priced differently. A switch is initiated from the active VM to a standby VM while accounting for various factors, e.g., cost, via optimization [26]. Different from Tier 2, Tier 3 may not lead to downgrading of performance, but it may lead to increased costs (due to the provisioning of redundant VMs).

3. Sample implementation of tier 1 on a 3D printer

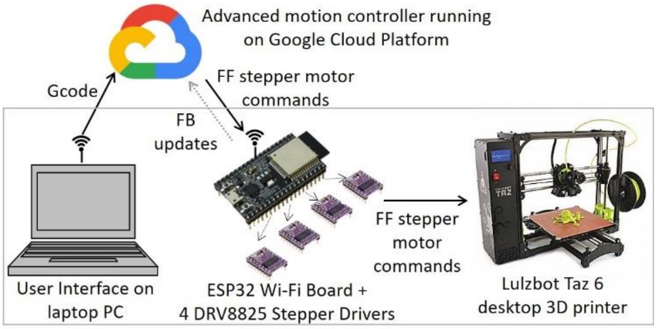
Advanced motion controller running on Google Cloud Platform
Gcode
FF stepper motor commands
FB updates
FF stepper motor commands
User Interface on laptop PC
ESP32 Wi-Fi Board + 4 DRV8825 Stepper Drivers
Lulzbot Taz 6 desktop 3D printer

Fig. 4 - Experimental set up.

Desktop 3D printers are suitable for initial testing of C-CNC because they have basic versions of CNC functionalities found in industrial-grade MMs. However, unlike industrial-grade MMs, desktop 3D printers often use open source hardware and software that permit ease of modifications needed to test C-CNC. In [14], a desktop 3D printer was controlled using FF-focused controller. Up to 54% reduction in printing time was demonstrated, without loss of printing accuracy, compared to the printer's default local controller. However, it was also shown that occasional long delays in data transmission over the Internet could cause sudden pauses in print that could mar the quality of printed parts. Similar problems can be expected in industrial-grade MMs controlled via C-CNC (e.g., in the form of vibration marks on machined surfaces). In this section, we show how switching smoothly between an advanced CC and a basic LC (as proposed in Tier 1 of our architecture) can help mitigate this issue.

Fig. 4 gives an overview of the experimental set up. It consists of a standard VM from Google Cloud Platform [27], and an ESP32 Wi-Fi board connected to four DRV8825 stepper motor drivers which are used to separately drive the X, Y, and Z-axes and E (extruder) motors of a Lulzbot Taz 6 3D printer located in Ann Arbor, Michigan. An advanced CC runs on the VM and calculates FF control signals in the form of commands for each stepper motor. The FF signals are delivered to the ESP 32 board (which serves as the LM) via the Internet. The CC receives FB updates on data delivery and printing status from the LM. The LM buffers the FF signals received from the CC and transmits them in real-time at 20 kHz to the 3D printer via the motor stepper drivers. The size of the buffer is limited by the memory available on the LM. In accordance with Tier 1 of our architecture, the LM runs a basic LC that takes over FF control of the printer if the buffer empties out due to long Internet transmission delays or server crashes. Note that switching from the CC to the LC due to emergency commands (e.g., feed override) as described in Tier 1 above, is not considered in this preliminary implementation. Details of the set up and experiments are discussed below.

3.1. Virtual Machine and Advanced Cloud Controller

Fig. 5 shows the contents of the advanced CC which runs on Google Cloud Platform's n1-standard-1 VM (having 1 vCPU, 3.75 GB memory and Linux OS). It consists of: (i) an indexer to select the Gcode line from which printing starts or continues; (ii) a jerk-limited motion command generator [14] which generates motion commands at 1 kHz for all axes of the printer; (iii) a vibration compensation algorithm [28] which convert the X- and Y-axis commands into modified commands (X' and Y') that minimize the vibration of the printer; and (iv) a stepper motor command generator which converts the respective axis commands into equivalent stepper motor commands at 20 kHz frequency. The stepper commands are placed in FF data packets, sequenced in chronological order and transmitted via user datagram protocol (UDP) to the LM (i.e., ESP 32 board) as detailed in [14]. FB data packets, comprising acknowledgement (ack) data and LC-to-CC switch data, are provided by the LM for purposes discussed in the following subsections. Note that, due to their relatively slow motions, the Z and E axis do not need jerk-limited motion nor vibration compensation [14].

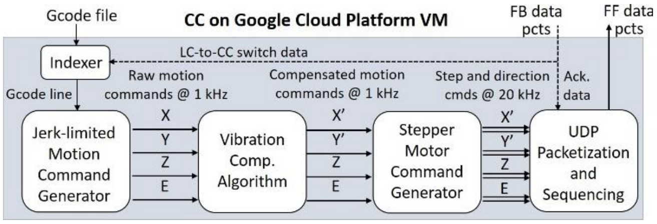
Gcode file
CC on Google Cloud Platform VM
FB data pcts
FF data pcts
Indexer
LC-to-CC switch data
Gcode line
Raw motion commands @ 1 kHz
Compensated motion commands @ 1 kHz
Step and direction cmds @ 20 kHz
Ack. data
X
Y
Z
E
X'
Y'
Jerk-limited Motion Command Generator
Vibration Comp. Algorithm
Stepper Motor Command Generator
UDP Packetization and Sequencing

Fig. 5 - Contents of advanced CC on the Google Cloud Platform.

3.2. Local Machine and Basic Local Controller

The contents of the LM (i.e., the ESP 32 board) are shown in Fig. 6. The FF data packets, containing the stepper motor commands, are received from the CC and stored in a first-in first-out (FIFO) buffer, following checks to ensure that the data has been accurately transmitted (as described in [14]). An ack signal is sent out as FB to the CC to indicate which packets have been received, and identify which packets need to be re-sent [14]. The buffer accommodates up to six seconds of motion commands to help mitigate transmission delays. During normal printing from the CC, stepper motor commands are fetched from the FIFO buffer in real-time at 20 kHz and delivered to the printer via the stepper motor drivers.

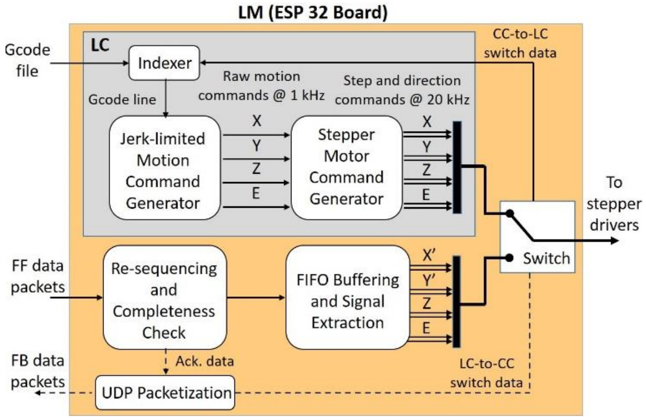
LM (ESP 32 Board)
LC
Gcode file
Indexer
CC-to-LC switch data
Raw motion commands @ 1 kHz
Step and direction commands @ 20 kHz
Gcode line
X
Y
Z
E
Jerk-limited Motion Command Generator
Stepper Motor Command Generator
To stepper drivers
Switch
FF data packets
Re-sequencing and Completeness Check
FIFO Buffering and Signal Extraction
X'
Y'
FB data packets
Ack. data
UDP Packetization
LC-to-CC switch data

Fig. 6 - Contents of LM and the Basic LC running on it.

The LM also contains a basic LC which serves as the backup for the CC, according to Tier 1 of the proposed architecture. The LC reads the same Gcode file (from an SD card) as the one read by the CC. An indexer determines the Gcode line number from which to resume printing. The LC uses the same jerk-limited and stepper motor command generators as the CC. However, because of the LM's limited computational resources, it cannot run the vibration compensation algorithm (which makes the CC advanced relative to the LC). The switch from the LC to CC and vice versa is governed by the switching logic described below.

3.3. Switching Logic

As shown in Fig. 6, the LM runs two parallel threads – one for fetching stepper motor commands from the CC and the other for generating stepper motor commands by the LC. The logic for switching smoothly between the CC and LC threads is described below. It consists of three scenarios.

3.3.1 Scenario 1 (CC in Control)

The system starts out with the CC in control. The LM receives FF data packets from the CC into the FIFO buffer until it is full before commencing the print job. Each FF data packet contains 1420 bytes of stepper motor commands. It also contains the following CC-to-LC switch data associated with the last stepper motor command in the packet: (i) the 4-byte Gcode line number; (ii) the numerical values of the X', Y', Z and E commands (16 bytes); and (iii) the speed (4 bytes). During printing from the CC, the packets are fetched out of the FIFO buffer and transmitted to the stepper motors. Meanwhile, the LC reads (from an SD card) the Gcode line corresponding to the last stepper motor command of the active FF packet so that it is always ready to take over printing from the CC.

3.3.2 Scenario 2 (switch from CC to LC)

If the FIFO buffer becomes empty before the print job is completed, the LM transfers the CC-to-LC switch data to the LC. The LC generates stepper motor commands that maintain continuity in position and speed to the very last stepper motor command sent out by the CC. Note that the switch from the CC to LC can occur at any position between the start and end positions associated with an active Gcode line. This allows the switch to occur as soon as the buffer is empty.

3.3.3 Scenario 3 (switch from LC to CC)

If connection to the CC is re-established before the print job is completed, the LC begins a process of switching back control to the CC. Re-establishment of connection to the CC is determined when the stale FF data packets (i.e., those that did not arrive before the buffer got empty) are received by the LM. The LC then determines a future Gcode line number, called the rendezvous line number (RLN), at which to switch to the CC. The RLN is currently set to N_{RLN} Gcode lines from the active Gcode line. Note that a RLN is not issued if the remaining number of Gcode lines to complete the print job is less than N_{RLN}; instead, the LC completes the remainder of the job.

The LC-to-CC switch data (405 bytes), which contains the RLN, is sent out via UDP connection to the CC in a FB data packet that also includes the ack signal described in Section 3.2. After sending out the LC-to-CC switch data, the LM flushes out the stale FF data packets from the FIFO buffer and fills it with the new stepper motor commands generated by the CC starting from the RLN. If the FIFO buffer is not full before the LC gets to the end of the Gcode line number immediately preceding the RLN, the switch to the CC is aborted and the process of switching to the CC is re-initiated with a new RLN. This process is repeated until the buffer is full before the LC gets to the end of the Gcode line immediately preceding RLN, in which case the CC takes over control starting from the RLN.

3.4. Results

The XYZ Calibration Cube with 20 mm sides [14] is used for testing the prototype implementation of Tier 1 described in the preceding section. The parameters used for slicing the cube are the same as presented in [14]. The speed and jerk limits are respectively 75 mm/s and 5000 m/s^3 for both the LC and CC. However, the acceleration limits are 10 m/s^2 for the CC and 1 m/s^2 for the LC. The difference in acceleration limits between the CC and LC is due to vibration compensation (as described in [14]). Leveraging this benefit, up to 54% faster printing was demonstrated in [14] using the CC compared to the LC, without sacrificing print quality.

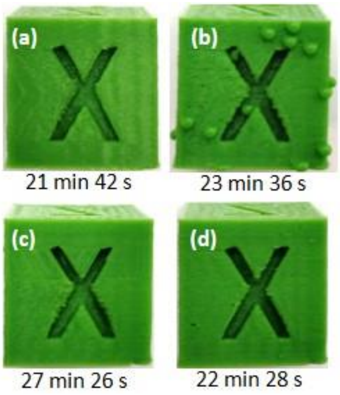

Fig. 7 - Prints of Calibration Cube using the CC: (a) w/o pauses (benchmark); (b) with pauses; (c) with pauses and switch from CC to LC but not vice versa; and (d) with pauses and switches from CC to LC and vice versa (proposed). Shown below each picture is the printing time.

However, it was also observed in [14] that the CC could experience long pauses due to Internet delays, potentially leading to poor print quality. This problem was seen when the CC was stationed a long distance away from the printer (e.g., in Australia). Though less likely, similar problems could occur (e.g., due to congestion or server crashes) when the CC is closer to the MM. To demonstrate this issue, 18 six-second delays were introduced artificially into the transmission from the CC to the LM, causing 18 pauses in print. Fig. 7 (a) shows pictures of the X-face of the Cube printed using the CC alone (without any pauses), i.e., the benchmark case. Fig. 7 (b) shows the effect of the pauses when printing from the CC. The surface quality of the block is marred by uncontrolled oozing of molten filament during the pauses. This highlights one of many possible undesirable effects of unreliability of C-CNC due to poor Internet QoS.

To resolve this issue, Tier 1 of the proposed architecture is utilized. The value $N_{\mathrm{RLN}} = 30$ is selected empirically based on the average data transmission rate from the CC to the LC and the average printing time of the Calibration Cube's Gcode by the LC. First, we consider a very conservative implementation of Tier 1 involving a switch from the CC to the LC as soon as the first pause occurs, without any switch back to the CC upon re-establishment of Internet connectivity. Fig. 7 (c) shows that acceptable surface quality of the printed cube is recovered but the printing time is 26.4% longer than that of the benchmark case in Fig. 7 (a). However, by implementing Tier 1 with the ability to switch from the LC back to the CC, the print quality is not only restored but the printing time is within 3.1% of that of the benchmark case.

A major reason for the acceptable quality of the Tier 1 implementation demonstrated in Fig. 7 is that continuity in position and speed were guaranteed during switching. Fig. 8 shows, as examples, snippets of the speed command profile around three of the 18 pauses. Notice the continuity in speed when switching from the CC to the LC and back to the CC. The continuity in speed occurs in all 36 switching instances.

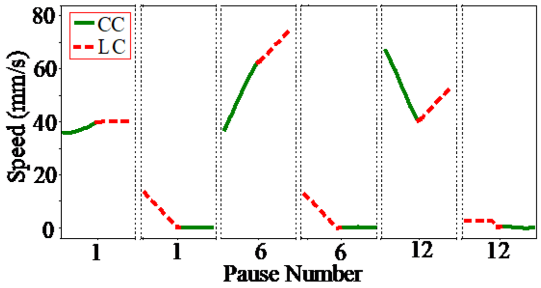

Fig. 8 - Snippets of speed profile around three of 18 pauses showing, in each case, continuity when switching from CC to LC and back to CC.

4. Research challenges

The proposed three-tier architecture presents interesting research challenges associated with guaranteeing safety and reliability without overly sacrificing the benefit of C-CNC over local CNC. Some of the research challenges emanating from the preliminary work on Tier 1 are discussed below.

4.1. Rendezvous Challenge

The contrast between the printing times of Fig. 7 (c) and (d) highlight the benefit to the overall performance of C-CNC to switch from the LC to the CC as soon as Internet QoS improves. However, the switch from the LC to the CC presents an interesting challenge. The LC decides a point in the future at which to pass over control to the CC. However, due to uncertainty in Internet delays, there is no guarantee that the CC will be ready to take over control at the appointed time. As a result, the LC and CC could go into a very long or even endless cycle of trying to schedule a rendezvous, thus diminishing the benefit of C-CNC over local C-CNC. In our preliminary experiments, we decided empirically that the rendezvous should take place at N_{RLN} = 30 Gcode lines ahead of the current Gcode line. Further research is needed to mathematically guarantee that a selected value of N_{RLN}: (i) cannot not lead to a very long or endless rendezvous cycle; and (ii) is not overly conservative, e.g., when dealing with Gcode lines that each represent long durations of printing.

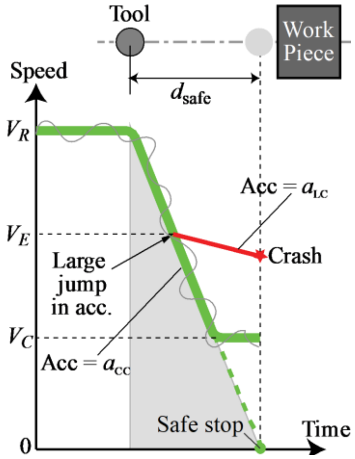

Fig. 9 - Example of C-CNC's reachability and smoothness issues.

4.2. Reachability Challenge

We assume that the CC is much more superior in performance to the LC to justify the need to use C-CNC. This means that there are tasks that the LC simply cannot do as well as the CC. This raises a question of how one can guarantee that the LC can always take over control from the CC. To illustrate this challenge, consider a scenario shown in Fig. 9, inspired by the preliminary work of Section 3. A CC is reducing the speed of a milling machine from rapid traverse speed V_R to cutting speed V_C at high deceleration a_{CC} (i.e., solid green line). However, during the deceleration, at speed VE halfway between V_R and V_C, an emergency stop button is pushed on the machine. The machine must immediately take a different course than that pre-planned by the CC to ensure a quick stop. Without guarantees that a new plan from the CC can be delivered on time over the Internet, the LC must step in to decelerate the machine.

However, with the LC's acceleration, $a_{\mathrm{LC}} << a_{\mathrm{CC}}$, the machine cannot come to a full stop within the safe stopping distance, d_{safe}, pre-planned by the CC, thus a crash will occur. Therefore, the CC must generate motion commands with guarantees that safe states are reachable by the LC (e.g., using reachability analysis [29]) without being overly conservative.

4.3. Smoothness Challenge

In our preliminary work, we achieved continuity in position and speed during switching. However, it is often desirable in CNC applications to also achieve continuity in acceleration. As discussed in Section 4.2 above, a major problem is that the acceleration limit of the CC is unreachable by the LC, leading to discontinuities in acceleration as depicted in Fig. 9. Moreover, the LC in our preliminary work does not perform vibration compensation but the CC does. Therefore, it is difficult to guarantee continuity between commands from the LC (which lack dynamic information) and those from the CC (which possess dynamic information –as shown by the wavy line in Fig. 9). In our preliminary work, we sidestepped this issue by switching from the LC to the CC at the end of Gcode lines where speed was typically (close to) zero for the Cube (see Fig. 8). However, it is of benefit to the overall performance of C-CNC to be capable of switching from the LC to the CC at any point during the motion of the machine. How to do this optimally is an open question.

5. Conclusion and future work

This paper has made a case for adding event-driven and/or slow periodic feedback to feedforward-focused C-CNC implemented over public Internet networks. It has also proposed a three-tier redundant architecture to ensure safe and reliable C-CNC and prototyped Tier 1 of the architecture successfully on a 3D printer. Some interesting research challenges introduced by the proposed architecture have been discussed, indicating the there is much left to be done to guarantee safe and reliable C-CNC. Future work will focus on tackling these and other challenges related to C-CNC.

References

[1] G. Adamson, L. Wang, M. Holm, and P. Moore, Cloud manufacturing–a critical review of recent development and future trends, *International Journal of Computer Integrated Manufacturing*, vol. 30, no. 4-5, 2017, pp. 347-380. https://doi.org/10.1080/0951192X.2015.1031704

[2] J. Krüger et al., Innovative control of assembly systems and lines, *CIRP Annals*, vol. 66, no. 2, 2017, pp. 707-730. https://doi.org/10.1016/j.cirp.2017.05.010

[3] L. Monostori et al., Cyber-physical systems in manufacturing, *CIRP Annals*, vol. 65, no. 2, 2016, pp. 621-641. https://doi.org/10.1016/j.cirp.2016.06.005

[4] X. Xu, Manufacturing machine 4.0 for the new era of manufacturing, *The International Journal of Advanced Manufacturing Technology*, vol. 92, no. 5-8, 2017, pp. 1893-1900. https://doi.org/10.1007/s00170-017-0300-7

[5] Y. Lu and X. Xu, Cloud-based manufacturing equipment and big data analytics to enable on-demand manufacturing services, *Robotics and Computer-Integrated Manufacturing*, vol. 57, 2019, pp. 92-102. https://doi.org/10.1016/j.rcim.2018.11.006

[6] B. Buckholtz, I. Ragai, and L. Wang, Cloud manufacturing: current trends and future implementations, *Journal of Manufacturing Science and Engineering*, vol. 137, no. 4, 2015, p. 040902. https://doi.org/10.1115/1.4030009

[7] Onshape. [Online]. Available: www.onshape.com . [Accessed: 08-Feb-2019].

[8] VisualCAMc. [Online]. Available: www.mecsoft.com/visualcamc . [Accessed: 08-Feb-2019].

[9] Machining Cloud. [Online]. Available: www.machiningcloud.com . [Accessed: 08-Feb-2019].

[10] A. Verl et al., An approach for a cloud-based manufacturing machine control, *Procedia CIRP*, vol. 7, 2013, pp. 682-687. https://doi.org/10.1016/j.procir.2013.06.053

[11] J. Schlechtendahl et al., Extended study of network capability for cloud based control systems, *Robotics and Computer-Integrated Manufacturing*, vol. 43, 2017, pp. 89-95. https://doi.org/10.1016/j.rcim.2015.10.012

[12] Z. Sang and X. Xu, The framework of a cloud-based CNC system, *Procedia CIRP*, vol. 63, 2017, pp. 82-88. https://doi.org/10.1016/j.procir.2017.03.152

[13] D. Tomzik and X. Xu, Architecture of a Cloud-Based Control System Decentralised at Field Level, *IEEE 14th International Conference on Automation Science and Engineering (CASE)*, 2018, pp. 353-358. https://doi.org/10.1080/0951192X.2015.1066861

[14] C. Okwudire et al., Low-Level Control of 3D Printers from the Cloud: A Step toward 3D Printer Control as a Service, *Inventions*, vol. 3, no. 3, 2018, pp. 56. https://doi.org/10.3390/inventions3030056

[15] J. P. Hespanha, P. Naghshtabrizi, and Y. Xu, A survey of recent results in networked control systems, *Proceedings of the IEEE*, vol. 95, no. 1, 2007, pp. 138-162. https://doi.org/10.1109/JPROC.2006.887288

[16] O. Givehchi, H. Trsek, and J. Jasperneite, Cloud computing for industrial automation systems—A comprehensive overview, *IEEE 18th Conference on Emerging Technologies and Factory Automation (ETFA)*, 2013, pp. 1-4. https://doi.org/10.1109/ETFA.2013.6648080

[17] A. Vick et al., Robot control as a service—towards cloud-based motion planning and control for industrial robots, *10th International Workshop on Robot Motion and Control (RoMoCo)*, 2015, pp. 33-39. https://doi.org/10.1109/RoMoCo.2015.7219710

[18] T. Hegazy and M. Hefeeda, Industrial automation as a cloud service, *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 10, 2015, pp. 2750-2763. https://doi.org/10.1109/TPDS.2014.2359894

[19] S. Mubeen et al., Delay mitigation in offloaded cloud controllers in industrial IoT, *IEEE Access*, vol. 5, 2017, pp. 4418-4430. https://doi.org/10.1109/ACCESS.2017.2682499

[20] C. Horn and J. Krüger, Feasibility of connecting machinery and robots to industrial control services in the cloud, *IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2016, pp. 1-4. https://doi.org/10.1109/ETFA.2016.7733661

[21] B. A. A. Nunes et al., A survey of software-defined networking: Past, present, and future of programmable networks, *IEEE Comm. Surveys & Tutorials*, vol. 16, no. 3, 2014, pp. 1617-1634. https://doi.org/10.1109/SURV.2014.012214.00180

[22] The Time Sensitive Networking Task Group. [Online]. Available: http://www.ieee802.org/1/pages/tsn.html. [Accessed: 07-Apr-2019].

[23] R. Gao et al., Cloud-enabled prognosis for manufacturing, *CIRP annals*, vol. 64, no. 2, 2015, pp. 749-772. https://doi.org/10.1016/j.cirp.2015.05.011

[24] Y. Altintas and D. Aslan, Integration of virtual and on-line machining process control and monitoring, *CIRP Annals*, vol. 66, no. 1, 2017, pp. 349-352. https://doi.org/10.1016/j.cirp.2017.04.047

[25] T. Stockhammer, Dynamic adaptive streaming over HTTP--: standards and design principles, *Proceedings of the second annual ACM conference on Multimedia systems*, 2011, pp. 133-144. https://doi.org/10.1145/1943552.1943572

[26] Z. Wu, C. Yu, and H. Madhyastha, CosTLO: Cost-Effective Redundancy for Lower Latency Variance on Cloud Storage Services, *NSDI*, 2015, pp. 543-557.

[27] Google Cloud. [Online]. Available: https://cloud.google.com/. [Accessed: 08-Feb-2019].

[28] M. Duan, D. Yoon, and C. Okwudire, A limited-preview filtered B-spline approach to tracking control—With application to vibration-induced error compensation of a 3D printer, *Mechatronics*, 2017. https://doi.org/10.1016/j.mechatronics.2017.09.002

[29] G. Reissig and M. Rungger, Abstraction-based solution of optimal stopping problems under uncertainty, *IEEE 52nd Conference on Decision and Control (CDC)*, 2013, pp. 3190-3196. https://doi.org/10.1109/CDC.2013.6760370