Proceedings of IMECE2007
2007 ASME International Mechanical Engineering Congress and Exposition
November 11-15, 2007, Seattle, Washington, USA

IMECE2007-42085

IMPROVED HAPTIC FIDELITY VIA REDUCED SAMPLING PERIOD WITH AN FPGA-BASED REAL-TIME HARDWARE PLATFORM

Kevin S. Sevcik
Dept of Mechanical Engineering
and Materials Science
Rice University
Houston, TX 77005
ksevcik@rice.edu

Emilie Kopp
National Instruments
Austin, TX
emilie.kopp@ni.com

Marcia K. O'Malley
Dept of Mechanical Engineering
and Materials Science
Rice University
Houston, TX 77005
omalleym@rice.edu

ABSTRACT

Impedance based haptic interfaces face inherent challenges in displaying stiff virtual environments. Fidelity of a virtual environment is enhanced by stiff virtual walls combined with low damping and passive behavior of the interface. However, the stiffness of virtual walls displayed on an impedance based interface is limited by the damping inherent in the controller, the sampling rate of the control loop, and the quantization of the controller's position. Attempting to display a stiffness larger than this limiting value destroys the passivity of the interface, leading to active controller behavior and eventually closed loop instability. We propose a method of increasing the fidelity of a PHANToM Premium 1.0 commercial haptic interface by controlling it via a Field Programmable Gate Array (FPGA) both alone and with a Real Time Operating System (RTOS) control system. This custom controller enjoys several benefits over the standard control achieved via a proprietary control card in a Multitasking OS, including reduced system overhead and deterministic loop rate timing. The performance of the proposed FPGA/RTOS controller compares favorably with the performance of an FPGA/Multitasking OS controller. The FPGA/RTOS controller achieves control loop rates an order of magnitude greater than that of the proprietary controller, allowing virtual walls to be displayed with greatly increased stiffnesses, while retaining the passivity and low damping of the PHANToM interface.

NOMENCLATURE

b	viscous damping
T	sampling period
f_c	Coulombic friction
Δ	encoder resolution

INTRODUCTION

Derived from the Greek word *haptesthai* (meaning to touch), haptics appropriately refers to stimulation of the human central nervous system in the form of touch sensations. This physical contact provides the ability for humans to perceive and manipulate the world around them; infants as young as four months integrate visual and haptic information when exploring the world [1].

In the past decade, the realm of physical perception has crossed over into the virtual world through implementation of robotic haptic devices. These machines can serve as a direct interface between human and computer, transmitting force feedback to convey physical traits of a virtual environment. Prior to the recent interest in haptic interfaces, humans typically interacted with computers through the click of a mouse or the press of a button and received information through visual or auditory displays, neglecting the sense of touch and feel. Haptic feedback, as an alternative form of communication between human and computer, is unique in that a bi-directional exchange of information in the form of mechanical energy occurs at a single continuous junction. This additional feedback lends greater fidelity and realism to simulated virtual environments, given an appropriately designed haptic device.

These haptic devices can be separated into two opposing classes: admittance devices and impedance devices. The fundamental difference in these classes lies in the nature of control and feedback between the user and the haptic device.

An admittance device receives a user's force applied to the device and changes position as appropriate to the applied force. This naturally mirrors the familiar physical laws of force, mass,

and acceleration. As an admittance device relies upon its own power for motion, and the user's strength for force display, it can accommodate a large physical structure without inconveniencing a user. In addition, admittance devices are capable of displaying very large stiffnesses, relying upon the added available mass for larger motors. However, admittance devices perform poorly at displaying low virtual masses, due to the practical lower bound on accurate force measurement. This, and the typical higher cost for admittance devices [2] has led many haptic researcher to focus on impedance devices.

An impedance haptic device is the operational inverse of an admittance device. Impedance devices receive a position from a user, and display a force based on that position, much as a typical motor controller would compare actual position with desired position to calculate the appropriate torque for the motor. This user-powered positioning necessitates a haptic device of low inertia and friction to preserve the illusion of free space, while simultaneously demanding powerful actuators to display large stiffnesses [2]. Often, the drive for lightweight and low friction reduces the cost and complexity of an impedance device. However, these traits also generate their own unique challenges in virtual environments.

Virtual Walls

In impedance devices, rendering a virtual environment entails computing appropriate interaction forces between the human operator and the virtual objects populating the environment according to the current position state vector. Upon collision with a virtual surface, mathematical representations of mass, stiffness, damping, etc. are conveyed to the user through the force-reflecting interface.

A universal facet of haptic rendering is the virtual wall. This fundamental building block is typically modeled as a simple, unilateral spring. Contact with a stiff virtual wall dictates the immediate and reversible switch between little or no impedance (free motion) to the sensation of infinite stiffness (rigid surface). The reaction force can be calculated with Hooke's Law: a linear product of virtual spring stiffness and the user's displacement of the virtual surface.

The fidelity of a rigid virtual wall is primarily dependent upon the virtual stiffness perceived by the human operator [3]. Yet despite the apparent simplicity of implementation, the rendering of a stiff virtual surface has become a ubiquitous challenge in the field of haptics. Because of the inevitably discrete nature of its implementation, a virtual spring always acts to generate or "leak" energy into the simulated system, thereby failing to capture the inherently passive (dissipative) nature of a physical spring of identical stiffness [3].

Previous research chronicling these energy leaks begins with their initial recognition by Kazerooni [4] as a consequence of sampled-data implementation. Further insight to this time-variance is provided by Gillespie and Cutkosky [5], who detail the ramifications of the zero-order hold (ZOH) and asynchrony of continuous and discrete traversal through the virtual wall threshold. Additionally, the work of Abbott and Okamura [6] provides explicit evidence that the quantized position signal necessary for impedance-based rendering can contribute to the non-passive nature of a virtual spring. Diolatti et al. [7] affirm this outcome but maintain that the energy generation is a result of a position signal that is both quantized and discretized, and not just the former.

In actuality, when rendering modest stiffness, the energy leaks are sufficiently dissipated by intrinsic device friction and thus remain transparent to the human operator. When the virtual wall model is applied to a haptic interaction, one can guarantee the system to be passive if and only if the virtual spring stiffness to be displayed does not exceed the minimum of two ratios: viscous damping to sampling rate and Coulomb friction to encoder resolution [8, 6].

$$K \leq \min\left(\frac{2b}{T}, \frac{2f_c}{\Delta}\right) \tag{1}$$

Creating a virtual wall that behaves passively has subsequently spawned extensive research dedicated to alleviating the limitations on achievable stiffness defined by Equation (1). Often times, the challenge can become a compromise or submission of one or more of the high performance haptic interaction standards recommended by Ellis et al. [9]. However, in recent years, many haptic researchers have proposed varying methods to meet this challenge.

Passivity Preservation

Many methods of maintaining passivity or compensating for energy leaks have been explored. For example, a conventional remedy for energy leaks that can degrade virtual stiffness perception is to simply compensate with more physical dissipation. Miller, Colgate and Freeman [10] promote this strategy, defining a function to determine the amount of inherent damping a haptic device should posses to sufficiently dissipate unwanted energy generation. However, excessive damping can compromise overall transparency, making free space no longer feel "free". The work of Colgate and Brown [8] foresees this caveat and suggests countering it with negative virtual damping. They admit, however, that this novel solution relies on exact cancellation of device friction with negative virtual damping. This precision is difficult to achieve in practice, and even then, the system is borderline passive.

An alternative solution to reduce the flow of system energy is the utilization of simulation software to avoid detection of leaks by the human operator. Colgate et al. [11] dispose of unwanted energy through the process of "virtual coupling," a scheme that modifies the simulated environment by means of a filtering mechanism to ensure that the passivity of the system is maintained. Hannaford and Ryu [12] embrace a similar approach of preserving system passivity, deploying an online passivity observer which appropriately prompts a passivity controller to adjust an adaptive and dissipative rendering element.

Instead of compensating for energy generation with physical or virtual dissipation, a more direct method of extending the passivity region of a virtual spring is to minimize the energy leaks stemming from the disparity between the continuous spring model and its discrete sampled-data

implementation. If we assume that we will leave the b and f_c terms unmodified in Equation (1), we are still left with the option of decreasing the sampling period or increasing the encoder resolution (depending on the limiting factor) to minimize creation of unwanted energy. As many haptic devices already dissipate much excess energy through Coulomb friction [7], efforts at improvement have primarily focused on increasing sampling rates.

Several means for increasing sampling rates have been explored. One solution is to simplify the rendering algorithm to alleviate the operating system of computationally heavy calculations [13], although this almost always sacrifices virtual scene complexity. Mahvash and Hayward [14] combine this tactic with that of the previously mentioned passivity controller in [5], resulting in an increase in servo rates of complex virtual environments.

Many haptic researchers have opted to streamline communication methods with a force-reflecting device by employing a real time operating system (RTOS), a type of event-driven OS that abides by a strict hierarchy of task execution [7, 15, 16]. Although this tactic has not yet been explicitly employed to upgrade the performance of passive virtual walls, RTOS's have long since been a boon to sampled-data control systems in general. The primary advantage of an RTOS lies in its ability to complete tasks deterministically on a time deadline, reducing jitter commonly associated with a pre-emptive multitasking OS (PMOS). However, the lack of pre-emption and its overhead can also allow an RTOS to achieve algorithmic control rates faster than those of a typical OS.

Further control rate gains can be realized through the use of a Field Programmable Gate Array (FPGA), an array of logic gates that can be configured at will to execute arbitrary digital operations. These operations can range from Boolean logic to digital counters and multiplication, all occurring as fast as the signal propagation permits [17]. The flexible nature of an FPGA allows for unrelated operations to occur completely in parallel; where a standard processor must perform multiple math operations sequentially, an FPGA can perform the operations simultaneously. In addition, the interface between an FPGA and a host operating system can be customized to the programmer's specifications to minimize the necessary communication. This speed and flexibility comes at a price, however, as the number of available logic gates on a given FPGA is limited, imposing restrictions on the ultimate amount of operations that can occur. Despites these limitations, FPGAs have been previously used as supplementary velocity estimators [18] and low power haptic controllers [19]. Galvan et al. implemented an FPGA based controller for the NASA JPL Force Reflecting Hand Controller of comparable complexity to commercial haptic devices [20].

This work examines the possible gains in sampling rate and virtual wall stiffness of an impedance haptic interface paired with an RTOS and FPGA based control system when compared to the same system controlled via a standard PMOS. The RTOS/FPGA combination itself provides two possible control schemes with the majority of control calculations occurring either on the RTOS host machine or the FPGA.

METHODS

In this paper we propose a method of increasing the fidelity of a PHANToM Premium 1.0 commercial haptic interface by controlling it via a Field Programmable Gate Array (FPGA) both alone and with a Real Time Operating System (RTOS) control system. This section will present the system architectures and the experiment design.

System Architectures

Three possible control schemes are examined in this work, though all rely on an FPGA acting as a custom device I/O solution. The most complex solution requires additionally that all control calculations occur onboard the FPGA, with only control parameters and data passed to the host machine, offering the greatest potential gains in sampling rates. Moving the calculations off the FPGA to a host machine running an RTOS still offers moderate sampling rate gains while greatly decreasing complexity and harnessing the floating point math capabilities of modern processors. Similarly, running a PMOS instead of an RTOS, while further reducing maximum possible sampling rates, offers simplified debugging and programming changes.

Multitasking Operating System

A control system based on a PMOS offers a reasonable approximation of the control scheme adopted by many commercial haptic interfaces. These interfaces rely on custom device I/O hardware connected to standard PC communication buses or ports. Device drivers and simulations run in a standard, interruptible PMOS. Programming an FPGA to mimic the function of custom device I/O hardware interfacing with the PMOS yields the communication path detailed in Figure 1.

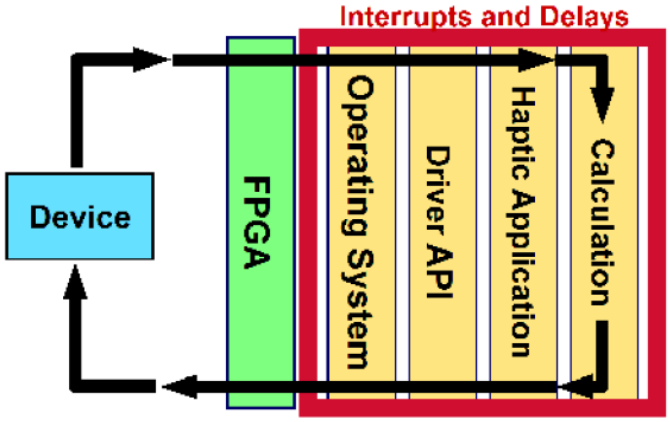


Figure 1. PMOS/FPGA communication path

This communication path simulates how a haptic device operating in a multitasking environment would function. Software running in a PMOS environment must contend with numerous hardware interrupts and the multitasking nature of the OS. Without strict control of other running software, a computationally intensive haptic application risks being starved of enough processor time to maintain a constant loop rate without missing updates. In addition, the application must deal

with interrupts caused by other hardware and software on the host machine. These interrupts halt all other software while they are being handled. Since the source of the interrupts is beyond the control of the haptic application, interrupts inject an essentially unpredictable amount of delay into the operation of the haptic application.

However, the application does benefit greatly from being executed on a modern processor with strong floating point capabilities. Floating point math allows arithmetic operations on numbers of greatly varying magnitude with little loss of precision. While more complicated and computationally expensive than integer math, this insures that an already quantized system suffers no further loss of information.

Real Time Operating System

An RTOS is similar to a PMOS in that it is a multitasking environment and it must still interface with a haptic interface through custom I/O hardware and device drivers. The fundamental difference is that an RTOS is designed to make it possible for the program to meet specified timing deadlines, through use of specialized scheduling software. The modified communication path can be seen in Figure 2.

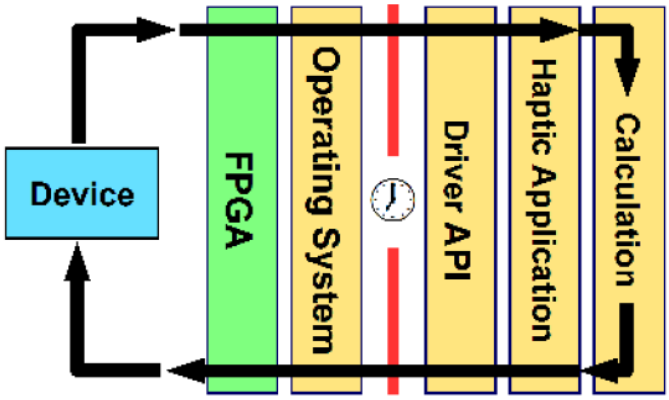


Figure 2. RTOS/FPGA communication path

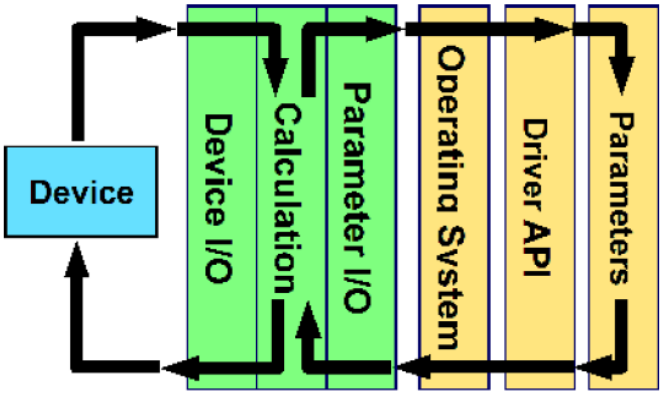


Figure 3. FPGA communication path

The fundamental difference is that the RTOS is capable of running deterministically, guaranteeing that properly developed software will run at a specified rate without missing any updates or suffering from interrupt induced jitter. The fact that an RTOS typical runs only the desired application and necessary support software helps to reduce the total load on the processor and increase possible loop rates.

Field Programmable Gate Array

A system based on the concept of performing all necessary calculation on an FPGA differs greatly in performance and implementation from the previous two control systems. The FPGA still acts as custom I/O hardware interfacing with the haptic device, however all necessary calculations are now carried out on the FPGA. This eliminates the possibility of any other software delaying the calculations for haptic feedback. The parallel nature of the FPGA fundamentally changes the nature of the communication path as shown in Figure 3.

With the FPGA calculating the appropriate feedback, a constant loop rate is guaranteed. The FPGA can, in fact, operate completely independent of input from the host machine, so long as parameter updates are not necessary. However, this does not come without a price. It is impractical to implement floating point math on an FPGA for all but the simplest haptic device. In practice, all math on an FPGA must be carried out as integers.

As all non-trivial haptic applications will require non-integers, a fixed point math algorithm must be developed in lieu of floating point math. This entails the use of a binary number that has a number of bits before and after the radix point. To convert a decimal number to a fixed point of this form one simply multiplies the original number by 2 raised to the power of the number of digits right of the radix. The answer is truncated and the remaining integer represents the original decimal number [21].

Fixed point math on the FPGA does have drawbacks. While basic mathematical operations remain largely the same, calculating any transcendental function would prove too computationally costly to be practical. The simplest solution to this would be to use a pre-calculated lookup table to acquire any necessary values. Additionally, a fixed point number lacks the dynamic range of a floating point number. This implies both a fixed precision and fixed upper and lower values for the number, so great care must be taken to maximize the precision while insuring that there will be no value overflows [21].

Experiment Design

A preliminary study was carried out with the RTOS/FPGA platform to investigate its possible benefits over the off-the-shelf configuration. This study was carried out on a Sensable PHANToM Premium 1.0, a popular three degree of freedom (DOF) commercial haptic device. We can calculate the performance of the PHANToM given the estimated relevant parameters in Table 1 [7].

These values combined with Equation (1) yield theoretical maximum stiffnesses of 10 N/m based on sampling rate and 2,612 N/m based on encoder resolution, while still maintaining passivity. This implies that the passivity criterion for the PHANToM is dominated by sampling rate, and therefore the PHANToM is an excellent candidate for a study of computational platforms.

Table 1. Estimated PHANToM parameters

Symbol	Value	Units
b	0.005	*N-s/m*
T	0.001	*S*
f_c	0.038	*N*
Δ	29.1	μm

A virtual environment was implemented on PMOS/FPGA, RTOS/FPGA, and FPGA platforms. The environment consisted of selectable pairs of opposing virtual walls in the *x*, *y*, and *z* planes, forming a cube shaped room. The FPGA device was a National Instruments PXI-7831R Intelligent DAQ device, housed inside a NI PXI-1083 chassis with a PXI-8186 Embedded Controller. National Instruments products were primarily chosen for ease of development, especially of the FPGA interface, which required no formal HDL language training.

In the PMOS and RTOS platforms, the FPGA was configured to act simply as an encoder counter, watch dog generator and D/A motor controller to interface with the PHANToM proprietary amplifier box. The virtual room was implemented on the NI embedded controller via NI's Labview V8.2 RTOS or in NI Labview, as appropriate. For the RTOS, the virtual room was controlled via a TCP/IP connection from a remote computer to the embedded controller, but all processing and data storage occurred on the embedded controller.

The FPGA platform was implemented using 32-bit integer math with a fixed point shift of 22 bits, yielding a precision of 2^{-22} cm. Trigonometric functions were implemented using 32 bit lookup tables to directly derive trigonometric values from encoder counts. However, due to lookup memory constraints of the FPGA, the functional workspace of the FPGA platform was about 90% the size of the other platforms, with invalid values returned at the extreme edges of the workspace. Otherwise, all necessary position calculation, collision detection, and force and torque calculation was handled on-board the FPGA with loop rates of up to 400 kHz, with the RTOS system acting merely as an interface and data acquisition system.

The authors examined the virtual wall responses of eight varying control platforms: a PMOS/FPGA platform at 1 kHz, an RTOS/FPGA platform at 1, 5, 10 and 20 kHz, and an FPGA platform at 20, 100, and 400 kHz. These will be referred to as MPOS-1, RTOS-1, RTOS-5, RTOS-10, RTOS-20, FPGA-20, FPGA-100, and FPGA-400.

To test the passivity of the virtual wall, weighted drop tests were performed at varying stiffnesses and sample rates. During a trial, the PHANToM was oriented with gravity acting parallel to the axis under test. Figures 4(a), 4(b), and 4(c) present the PHANToM oriented for tests in the *x*, *y*, and *z* axes, respectively. An adjustable weight was positioned at the end of the PHANToM in place of a stylus to simulate a human touch. This weight was adjusted for each orientation such that the end of the linkage exerted .15 N on a scale.

The virtual environment under test in each trial consisted of a pair of virtual walls oriented normal to gravity and the axis under test. These walls were situated 2 cm from the zero position of the PHANToM and the stiffnesses were varied from 5 N/m up to 40,000 N/m where possible.

The PHANToM itself was supported in the zero position by a digitally controlled solenoid release. At the start of the trial, the solenoid would release the PHANToM and the control platform would record the Cartesian coordinates of the weight at 1kHz for 10 seconds to provide ample data to examine the platform's response while resting on the wall. Three trials were run in each direction, at each stiffness, for each platform.

EXPERIMENTAL RESULTS

Figure 5 presents the responses of two RTOS-20 trials in the Y-direction with stiffnesses of 250 N/m and 5000N/m. The responses have been cropped to 2 seconds for clarity, but it is easy to see the empirical difference between the passive response at 250 N/m and the active response at 5000 N/m.

The transition from passive response to active response is a small change, and would be difficult to detect simply from visual inspection of a full size plot as demonstrated by Figure 6. Three active responses of the RTOS-20 are presented with varying virtual wall stiffnesses. The amplitudes of the responses are clearly the same, but the response at 2500 N/m is also clearly more active than at 500 N/m.

(a) X-Axis Orientation (b) Y-Axis Orientation (c) Z-Axis Orientation

Figure 4. PHANToM drop test orientations used in experiments

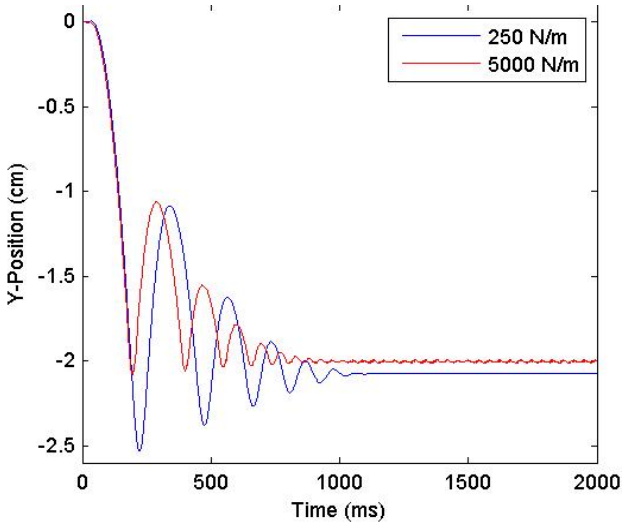


Figure 5. RTOS-20 passive-active behavior comparison

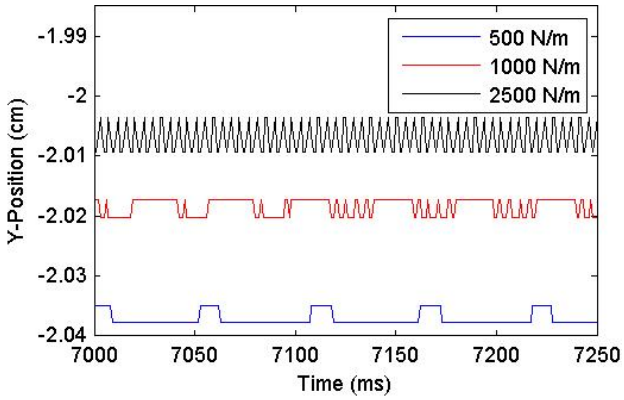


Figure 6. RTOS-20 passive-active behavior transition

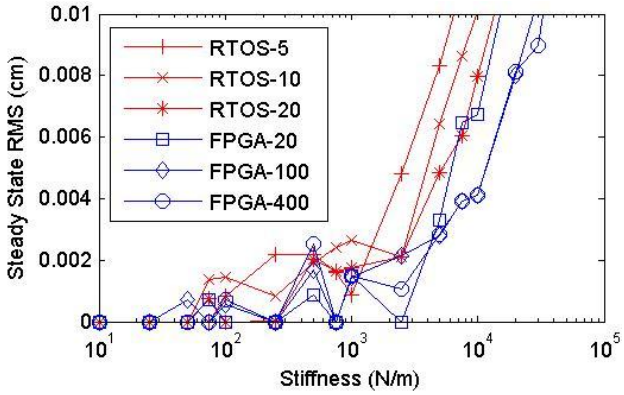


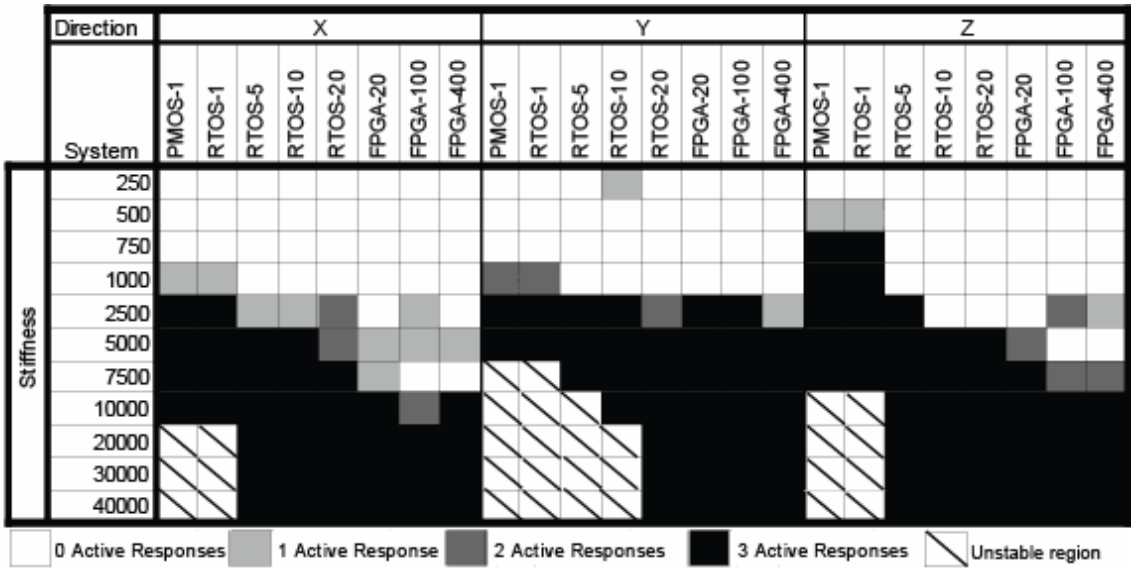
Figure 7. Y-Axis RMS values for RTOS and FPGA

To compare the passivity of the various systems at varying stiffnesses, the steady state response of the system between 7000 ms and 10000 ms is analyzed. The root mean square (RMS) of the data around the minimum steady state value is recorded as a rough measure of the excess energy in the system. Passive response was characterized by an RMS less than 0.003 cm, or the resolution of the encoders.

The RMS data provides insight into where the passive to active transition occurs. Table 2 presents the total active responses out of the three trials at each data point. This indicates the maximum passive stiffness for each platform.

RMS values in the FPGA and RTOS platforms followed different trends, as displayed in Figure 7. The RTOS and FPGA platforms both initially transitioned to active behavior gradually, but the FPGA platform did not increase as rapidly in active behavior. In addition, the RTOS-20 and FPGA-20 values track closely, as do the FPGA-100 and FPGA-400 values.

Table 2. RMS active responses over 0.003 cm (threshold for passivity)



DISCUSSION

The experimental results show that increasing sampling rates through the use of an RTOS/FPGA platform can greatly increase the range of achievable stiffnesses of an impedance based haptic device while maintaining passivity. The maximum achievable stiffness can clearly be seen to increase as the sampling rate increases, which demonstrates the benefit to be gained by employing RTOS platforms for haptic control, even with more modest increases in sampling rate.

The FPGA platform clearly equaled or exceeded the performance of RTOS platform. However, at high stiffnesses, the FPGA-100 and FPGA-400 platforms exhibited very similar performance. This suggests that the sampling rate increase from 100kHz to 400kHz has little effect on the response of the system. One possible explanation is that the 400kHz loop rate has exceeded the bandwidth of the device actuators or the PHANToM amplifier and the motors and/or amplifiers are now the limiting factor. However, another possibility is that past 100kHz, the sampling rate has ceased to be the limiting factor, and the encoder resolution is now what limits the achievable stiffness in the PHANToM. A sampling rate of $10\mu s$ would bring the stiffness limit to 1000 N/m, which is much closer to the Coulomb stiffness limit of 2,612 N/m.

The possible benefits of FPGA interfaces to the haptic community are great in spite of the challenge of their application. This proof of concept that a haptic environment can be completely simulated on a modest FPGA opens many doors. It could be configured to act as an adaptable interface to many different haptic devices. If the FPGA is capable of handling all translation to and from Cartesian space, virtual environments and haptic enabled programs could be coded in pure Cartesian space with the FPGA acting as translator to whatever haptic interface is required, greatly simplifying the development of new programs. More advanced FPGA devices could be configured to deterministically run fast local models to control haptic interfaces, leveraging the greater storage and floating point abilities of a host computer to update the those models as necessary, greatly improving the fidelity of the haptic environment. For applications that involve greater complexity in their graphical models and rendering algorithms, porting of some of the data acquisition and communication between the device and software to FPGA still enables the system to realize increased update rates, thereby improving fidelity.

CONCLUSIONS

This paper investigates a method for improving haptic fidelity by increasing achievable stiffnesses of virtual walls on a PHANToM 1.0 haptic interface via increasing the sampling rate. The proposed RTOS/FPGA and FPGA platforms greatly increased maximum achievable stiffnesses on a commercial haptic display by enabling efficient computation, fast communication between hardware and software, and guaranteed sampling rates. The results validate the proposed platforms as a feasible method of increasing the fidelity of haptic virtual environments via increased sampling rate, with achievable virtual wall stiffnesses exceeding those on the off-the-shelf PHANToM computational platform by an order of magnitude.

ACKNOWLEDGMENTS

The authors acknowledge National Instruments for providing hardware for the experiments.

REFERENCES

[1] Rochat, P., 1989. “Object manipulation and exploration in 2- to 5-month-old infants”. *Developmental Psychology,* **25**(6), pp. 871-884.

[2] Salisbury, K., Francois, C., and Barbagli, F., 2004. “Haptic rendering: Introductory concepts”. *IEEE Computer Graphics and Applications,* **24(**2), pp. 24-32.

[3] Colgate, J., Grafing, P., Stanley, M., and Schenkel, G., 1993. “Implementation of stiff virtual walls in force-reflecting interfaces”. *Proc. IEEE Virtual Reality Symp.,* pp. 202–208.

[4] Kazerooni, H., 1993. “Human induced instability in haptic interfaces”. *Proc. ASME Winter Annual Meeting.,* **55**(2), pp. 851-856.

[5] Gillespie, B., and Cutkosky, M., 1996. “Stable user-specific rendering of the virtual wall”. *Proc. ASME IMECE*, **58**, pp. 397–406.

[6] Abbott, J. J., and Okamura, A. M., 2005. “Effects of position quantization and sampling rate on virtual-wall passivity”. *IEEE Trans. Robot.*, **21(**5), pp. 952–964.

[7] Diolaiti, N., Niemeyer, G., Barbagli, F., and Salisbury, J.K., Jr., 2006. “Stability of Haptic Rendering: Discretization, Quantization, Time Delay, and Coulomb Effects”. *IEEE Transactions on Robotics,* **22**(2), pp. 256-268.

[8] Colgate, J., and Brown, J.,1994. “Factors affecting the z-width of a haptic display”. *Proc. IEEE Int. Conf. Robot. Autom.*, pp. 3205-3210.

[9] Ellis, R., Ismaeil, O., and Lipsett, M., 1996. “Design and evalutaion of a high-performace haptic interface”. *Robotica*, **14**(3), pp. 321-327.

[10] Miller, B., Colgate, J., and Freeman, R., 2004. “On the role of dissipation in haptic systems”. *IEEE Trans. Robot.,* **20**(4), pp. 768–771.

[11] Colgate, J., Grafing, P., Stanley, M., and Schenkel, G., 1993. “Implementation of stiff virtual walls in force-reflecting interfaces”. *Proc. IEEE Virtual Reality Symp.,* pp. 202–208.

[12] Hannaford, B., and Ryu, J., 2001. “Time-domain passivity control of haptic interfaces”. *Proc. IEEE Int. Conf. Robot. Autom*, pp. 1863–1869.

[13] Hayward, V., and Astley, O., 1996. “Performance measures for haptic interfaces”. *Proc. 1996 Robotics Research: 7th Int’l Symp.,* pp. 195-207.

[14] Mahvash, M., and Hayward, V., 2005. “High-fidelity passive force reflecting virtual environments”. *IEEE Trans. Robot.,* **21**(1), pp. 38–46.

[15] Mahvash, M., and Hayward, V., 2005. “Haptic Rendering of Cutting: A Fracture Mechanics Approach”. *Haptics-e. [Online],* **2**(3), Available: http://www.haptics-e.org

[16] McKnight, S., Melder, N., Barrow, A.L., Harwin, W. S., Wann, J. P., 2005. *“*Perceptual Cues for Orientation in a Two Finger Haptic Grasp Task*”. Proc. IEEE WHC'05,* pp. 549-550.

[17] National Instruments, 3 Jul. 2005. “Introduction to LabVIEW FPGA”. *LabVIEW 8 FPGA Module Training,* Retrieved from: zone.ni.com/devzone/cda/tut/p/id/3555/

[18] Çavusoglu, M.C., Feygin, D., and Tendick, F., 2002. “A Critical Study of the Mechanical and Electrical Properties of the PHANToM Haptic Interface and Improvements for High Performance Control”. *Presence,* **11**(6), pp. 555-568.

[19] Holbein, M. and Zelek, J.S., 2005. "A FPGA Haptics Controller". *Proc. IEEE WHC'05,* pp. 588-589.

[20] Galvan, S., Botturi, D., and Fiorini, P., 2006. “FPGA-based Controller for Haptic Devices”. *2006 IEEE/RSJ Int'l Conf Intelligent Robots and Systems,*pp.971-976.

[21] Kraeling, M.B., 1996. “Fixed-point math in time-critical C applications”. *Proc.WESCON/96,* pp. 587-593.