PHANSIM : A SIMULINK TOOLKIT FOR THE SENSABLE PHANTOM HAPTIC DEVICES

Alireza Mohammadi, Mahdi Tavakoli, Ali Jazayeri
Department of Electrical and Computer Engineering
University of Alberta
Edmonton, AB, Canada
alireza3@ualberta.ca, tavakoli@ece.ualberta.ca, ali.jazayeri@ualberta.ca

ABSTRACT

The PHANToM® devices (SensAble Technologies Inc., MA, USA) provide the users in industry and academia with an opportunity for research and education in virtual reality, haptics, robot motion control and teleoperation. Traditionally, one has to develop C/C++ codes using the OpenHaptics® software development kit (SDK) in order to use these devices. The PHANSIM Toolkit is an academic/non-commercial Simul- ink toolkit for real-time motion control and teleoperation of the PHANToM haptic devices. This toolkit facilitates using the PHANToM haptic devices in Simulink. The usefulness of this toolkit is demonstrated through two illustrative experiments using two different types of PHANToM products, namely the Omni and the Premium 1.5 models.

Keywords: PHANToM haptic devices, Simulink, robotics, teleoperation.

INTRODUCTION

The word *haptic*, from the Greek origin *haptikus*, means of/related to the sense of touch. Haptic technology enables the users to touch and manipulate remote or virtual objects in remote or virtual environments. This technology has found applications in a wide variety of areas such as video games, medical training, scientific visualization, computer animation, remote vehicle and robot control, and medical rehabilitation. The SensAble PHANToM haptic devices [1] are among the most popular commercial haptic devices that provide the users with an opportunity for research and education in haptic technology and its applications. The PHANToM product line includes a large variety of haptic devices, from the Premium models with high-precision, large workspaces and high forces to the Omni model that is one of the most cost-effective haptic devices available in the market. These haptic devices have been used in various applications that provide the user with the sense of touch such as computer games [2], surgical simulators [3], virtual rehabilitation excercise systems [4] and teleoperation systems [5]. Robotics education is another area where these haptic devices, especially the affordable PHANToM Omni, have been used effectively [6].

The OpenHaptics SDK [7], provided by the SensAble Technologies Inc., enables the users to develop C/C++ programs to work with the PHANToM haptic devices. However, it does not provide easy access to the device inputs and outputs, easy integration with external hardware such as cameras, and advanced mathematical functions such as matrix operations and filtering tasks. On the other hand, MATLAB/Simulink software package [8], which supports external hardware integration and a large variety of mathematical functions, has been used to develope versatile real-time interface for motion control of different robots such as KUKA manipulators [9]. A noncommercial interface has also been developed for the
PHANToM Omni [10]. However, it requires dismantling the haptic device and using an additional hardware system, i.e. dSPACE system, in order to use this interface. This serves as the motivation to develop an academic/non-commercial Simul- ink toolkit for the PHANToM haptic devices.

The PHANSIM Toolkit uses C/C++ S-functions along with the OpenHaptics toolkit to make an interface that provides the users with access to the PHANToM torque/force inputs, the Cartesian pose (position and orientation) of the gimbal and the joint angles of the device in the Simulink environment. The toolkit supports the operation of a single haptic device as well as the teleoperation of a master-slave system consisting of two haptic devices. This toolkit enables the users to implement and test their designed controllers on the PHANToM devices in a fast and easy way. The toolkit can be downloaded from the webpage: http://www.ece.ualberta.ca/~alireza3/Research.
html.

In this paper, we will introduce the PHANSIM Toolkit and its capabilites. We will explain how this toolkit works. Also, we will provide an overview on the Simulink blocks which are provided in the PHANSIM Toolkit library. Finally, we show the usefulness of the toolkit by two illustrative experiments, namely a circle drawing task using an Omni model and a bilateral teleoperation task using an Omni model as the master device and a Premium 1.5 model as the slave device.

OVERVIEW OF THE PHANSIM TOOLKIT

The PHANSIM Toolkit builds up a Simulink interface/block on top of the OpenHaptics Toolkit. The OpenHaptics HDAPI (Haptic Device Application Programming Interface) functions included in the OpenHaptics Toolkit enable the programmers to access and manipulate the low-level signals of the haptic devie (e.g., joint torque commands and joint position readings) using C/C++ [10]. One important component of the HDAPI is the scheduler component which enables the developer to communicate with the underlying servo-loop thread without using platform specific synchronization and thread related system calls [10]. In order to create an interface for the haptic device, which enables the users to set the input force/torque of the haptic device and to read the device states in the Simulink environment, we have used C/C++ S-functions to access the OpenHaptics HDAPI functions from Simulink. Figure 1 depicts how the Simulink communicates with the physical device by using the toolkit S-functions and the OpenHaptics HDAPI functions.

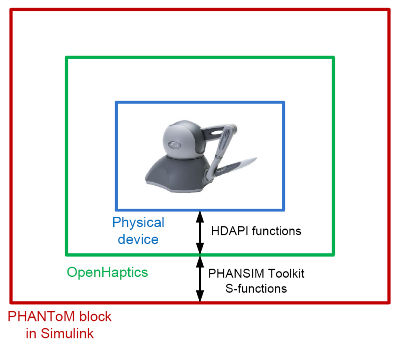

Fig.1: PHANSIM Toolkit hierarchy.

In order to work with a haptic device using the OpenHaptics Toolkit, the HDAPI functions should be called according to the following pattern (the readers are referred to the OpenHaptics Toolkit guide for a thorough explanation) [10]:

1. The device should be initialized.
2. The force outputs should be activated and the device scheduler function should be defined and started.
3. The force outputs should be disabled and the scheduler should be cleaned up.

The PHANSIM Toolkit S-functions use the above pattern to invoke the HDAPI functions and manipulate the haptic device. Figure 2 depicts the flow chart of the PHANSIM interface. As it can be observed, when the user starts the Simulink model, the haptic device will be initialized, device force outputs will be activated and the device scheduler function will be started. The scheduler function is called every 1 millisecond and applies the force or torque inputs, which are provided by the Simulink S-functions, to the device and reads the states (e.g., joint angles and gimbal position) of the device, which are reported to the Simulink S-functions. The Simulink should be able to update the inputs provided for the haptic device with a rate faster than or equal to 1 kHz if a smooth motion is required. The rate at which the inputs of the device are updated in the Simulink can be set through the Configuration Parameters dialog box of the Simulink model. In fact, one can consider the scheduler function as a sample-and-hold device which reads the inputs and reports the outputs at a rate of 1 kHz. When the user terminates running the Simulink model, the haptic device scheduler will be stopped and the device force outputs will be disabled. In short, the PHANSIM Simulink interface initialzes the device, provides input for the scheduler function, reads the device output and stops the device at the end by using the HDAPI functions.

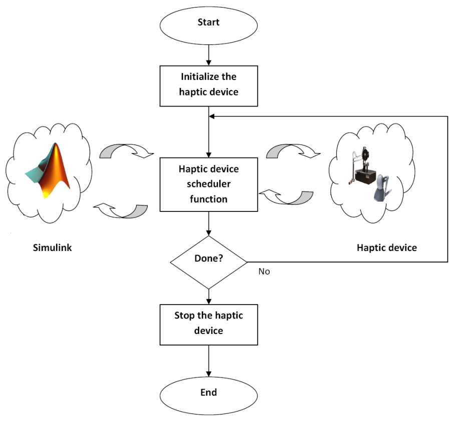
Start
Initialize the haptic device
Haptic device scheduler function
Simulink
Haptic device
Done?
No
Stop the haptic device
End

Fig.2: Flow chart of the PHANSIM interface.

The PHANSIM library consists of the following blocks:

- **PHANToM Block** This block can be used when only one haptic device is used. Force or torque inputs may be applied to the device. The user can determine the domain of the inputs (joint-level vs. Cartesian-level) in the block dialog box. Figure 3 depicts the schematic diagram of the PHANToM haptic device. Gimbal coordinates (x, y and z) and angles (q_1, q_2 and q_3) and the angles of the first three actuated joints (θ_1, θ_2 and θ_3) can be read from the device. If more than one haptic device is connected to the PC, the user can choose the desired device by typing its name in the block dialog box. The device name is the name by which the SenSable's PHANTOM Test program recognizes the haptic device. Figure 4 shows the PHANTOM Block.

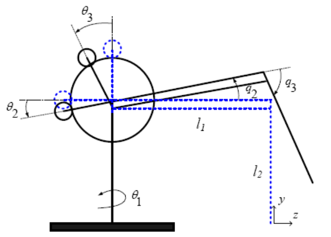

Fig.3: PHANToM schematic diagram.

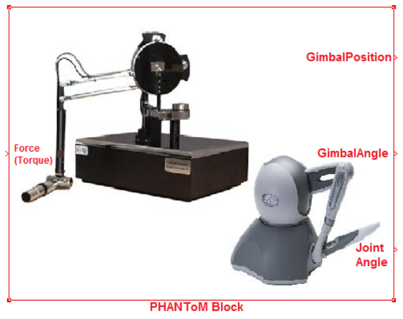

Fig.4: PHANSIM Library: PHANTOM Block.

- **PHANTOM Teleoperation Block** This block can be used when two haptic devices are used simultaneously. Again, force or torque inputs may be applied to the devices; the user can determine the type of inputs in the block dialog box. Gimbal coordinates and angles and the angles of the first three actuated joints of the robots can be read. The user can identify each device by typing its name in the block dialog box. Figure 5 shows this block.

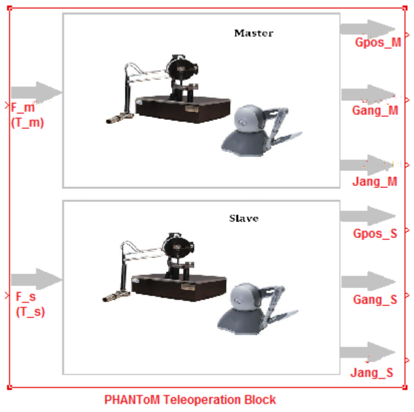

Fig.5: PHANSIM Library: PHANTOM Teleoperation Block.

- **PHANTOM Clock Generator Block** Since Simulink's simulation time is not representative of the actual time elapsed during an experiment, a block is needed to synchronize the simulation with the actual time. The PHANTOM Clock Generator Block generates a real-time clock signal by reading the CPU time. This block can be used to generate real-time signals. For instance, if we want to record the time passed during a teleoperation task, we should use this block's output to record the time. Figure 6 shows this block.

Fig.6: PHANSIM Library: PHANTOM Clock Generator Block.

EXPERIMENTS

This section illustrates the effectiveness of the toolkit by two experiments: a circle drawing task and a bilateral teleperation task. In the experiments, a PHANToM Omni haptic device was connected to a PC through the IEEE 1394 port and a PHANToM Premium 1.5 haptic device was connected to the same PC via the parallel port (EPP) interface. The following software programs were installed on the PC:

- Microsoft Windows® XP (service pack 2),
- MATLAB® R2009a (32-bit version),
- OpenHaptics® 3.0 academic edition.

A.Circle Drawing Task

A PHANToM Omni haptic device was used to draw a circle with a radius of 30^{mm} on a horizontal plane. Proportional-derivative (PD) control laws were used as control schemes at each joint with proportional and derivative gains equal to 0.25 and 0.7, respectively. Through the PHANTOM Block, a force input (i.e., PD controller output) was applied to the device, and the Cartesian coordiantes of the device gimbal were read. The reference trajectory was:

$$x(t) = 30\sin(\frac{\pi}{2}t),\ y(t) = 0,\ z(t) = 30\cos(\frac{\pi}{2}t) \qquad (1)$$

The Simulink PID controller block from *Simulink Extras/Additional Linear library* was used to implement the PD control law. Note that the output of the PHANToM Clock Generator Block is used to synchronize the *sine wave* sources with the actual time which is read from the PC's CPU. Figure 7 depicts the gimbal's x and z coordinates time-histories and the final trajectory of the device gimbal.

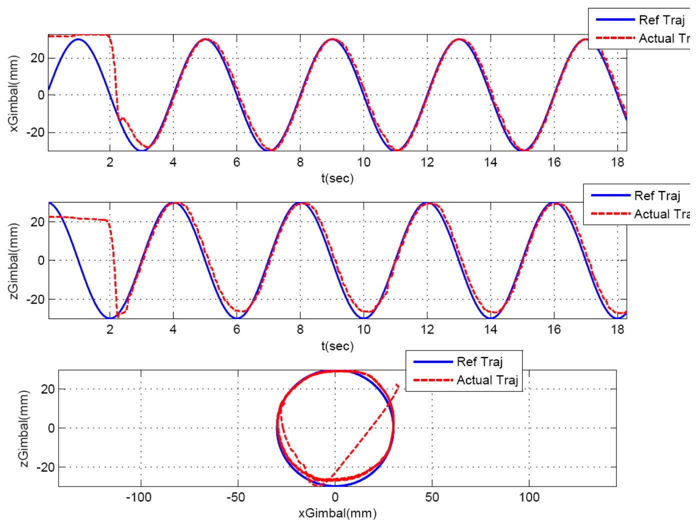

Fig.7: Circle drawing task.

B.Bilateral Teleoperation Task

In this experiment, a 1-DOF bilateral teleoperation task was done using two haptic devices. The PHANToM Omni haptic device was used as the master device and the PHANToM Premium 1.5 was used as the slave device. Through the PHANToM Teleoperation Block, torque inputs were applied to the first joints of the haptic devices and the angles of the first joints of the devices were read. Two proportional-derivative (PD) controllers were used at the master and the slave sides with proportional and derivative gains equal to 0.25 and 0.7, respectively.

Similar to the previous experiment, the Simulink PID controller block from *Simulink Extras/Additional Linear library* was used to implement the PD control law. Figure 8 depicts the time-history of the first joint angles of the PHANTOM OMNI and the PHANTOM Premium 1.5, respectively.

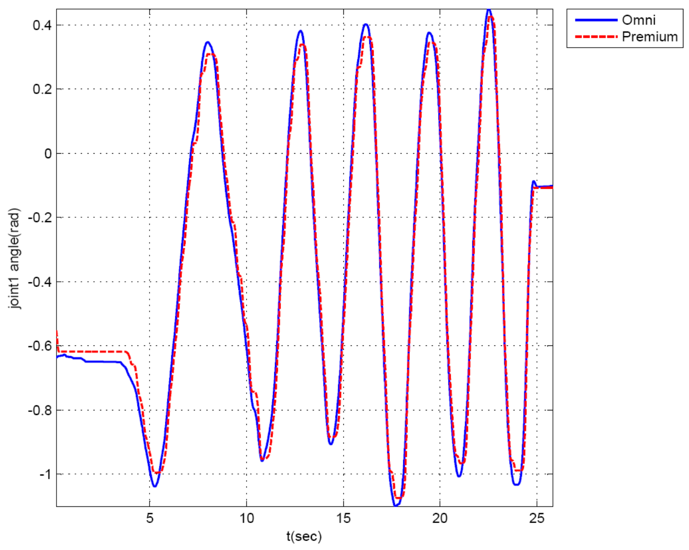

Fig.8: Bilateral teleoperation.

CONCLUSIONS

An academic/noncommercial Simulink toolkit for easy interfacing with SensAble PHANToM haptic devices was introduced in this paper. The toolkit can serve as a good and quick means for research and education purposes. It relieves the users from cumbersome hand coding in C/C++. The users can develop and implement their control systems in Simulink and test them on PHANToM haptic devices with ease and speed.

ACKNOWLEDGMENTS

This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC).

REFERENCES

[1] *SenSable Technologies Corporation*, http://www.sensable.com/.

[2] Fyans, A. C., and McAllister, G., 2008, "Creating games with feeling," *Proc. Int. Conf. on Computer Games: Artificial Intelligence and Mobile Systems*, Las Vegas, NV, pp. 94–98.

[3] Trier, P., Noe, K. O., Sorenson, M. S., and Mosegaard, J., 2008, "The visible ear surgery simulator," *Proc. Conf. on Medicine Meets Virtual Reality*, Long Beach, CA, pp. 523–525.

[4] Guo, S. X., Song, Z. B., and Ren, C. C., 2009, "Development of upper limb motor function training and rehabilitation system," *Proc. IEEE Int. Conf. on Mechatron. Autom.*, Changchum, China, pp. 931–936.

[5] Tzafestas, C., Velanas, S., and Fakiridis, G., 2008, "Adaptive impedance control in haptic teleoperation to improve transparency under time-delay," *Proc. IEEE Int. Conf. on Robot. Autom.*, Pasadena, CA, pp. 212–219.

[6] *Medical robotics and computer-integrated intervention at the University of Alberta-EE464*, http://www.ece.engineering.ualberta.ca/en/Undergraduate/Courses.aspx.

[7] *OpenHaptics Toolkit, The SenSable Technologies Inc., USA*, http://www.sensable.com/products-openhaptics-toolkit.htm/.

[8] *MATLAB and Simulink for Technical Computing, The MathWorks Inc., USA*, http://www.mathworks.com/.

[9] Chinello, F.,Scheggi, S., Morbidi, F., and Prattichizzo, D., 2010, "KCT: a MATLAB toolbox for motion control of KUKA robot manipulators," *Proc. IEEE Int. Conf. on Robot. Autom.*, Anchorage, Alaska, 2010, pp. 4603–4608.

[10] Eriksson M., and Wikander, J., 2010, *A haptic interface using Matlab/Simulink*, Sweden, Stockholm: Mechatronics Laboratory, Machine Design, KTH.