DESIGN OF A SPATIAL LINKAGE HAPTIC INTERFACE

John Ryan Steger  Ronald M. Chung  Kevin Lin  H. Kazerooni

Department of Mechanical Engineering
University of California, Berkeley
Berkeley, CA 94720

ABSTRACT

This paper describes the mechanical and electrical design of a compact high fidelity desktop haptic interface that provides three-degree-of-freedom point-force interaction through a handheld pen-like stylus. The complete haptic device combines a spatial linkage, actuation, power amplification, and control electronics in a standalone package with a footprint similar to that of a notebook computer (33cm x 25cm x 10cm). The procedure used to design the statically balanced spatial linkage is explained and both an inexpensive lightweight plastic version and a high stiffness, high strength, aluminum and stainless steel version are presented. The theory and implementation of sinusoidal encoder interpolation and sinusoidal servo-motor commutation used to achieve high-fidelity haptic simulation is covered for two versions of electronic control hardware: custom hardware based on a digital signal processor (DSP) and an off-the-shelf design based on an embedded PC.

KEYWORDS

Haptic interface; spatial linkage; static balance; sinusoidal interpolation; sinusoidal commutation.

OBJECTIVE

Our goal was to create a portable self-contained haptic device for precision finger-scale interaction that could enable point force interaction over a three-dimensional transnational work volume. The resulting device is a desktop unit intended to provide a haptic interface to PC based virtual environments (Fig. 1). Through the use of a unique kinematic linkage [1], we sought to create a haptic device with capabilities equal to or better than those of commercially available haptic interfaces.

Because the purpose of a haptic device is to simulate the forces generated through interaction with physical objects, performance is typically measured by the level of tactile realism achieved. Guidelines for obtaining a realistic haptic simulation include multiple degrees of freedom, human matched force capability, low friction, near-zero backlash, and gravitational counterbalancing [2]. The present joystick design addresses these points and offers exceptionally smooth force output, low inertia resulting in high motion bandwidth, and high motion resolution. Our design goal is to develop a single unit stand-alone desktop haptic interface device that combines high performance with portability and low manufacturing cost.

BACKGROUND

Through application of our patented kinematic linkage architecture [3], we sought to create a haptic device with capabilities comparable in many respects to those currently available in commercial haptic interfaces like the PHANToM Haptic Interface, produced by SensAble Technologies [4]. The performance specifications of the original PHANToM (Model: “Premium 1.0”) set the following targets for realistic simulation of precision finger-scale manipulation: less than 0.1 N static back-drive friction, inertia such that the user perceives unbalanced weight of 0.2 N or less, maximum stiffness of 35 N/cm, peak and continuous end effector forces of 10 N and 1.5 N respectively, and end effector spatial resolution of at least 400 dpi (98 µm) [4]. Like many of its predecessors, the PHANToM makes use of serial linkage elements as part of the force transmitting mechanism. A purely serial mechanism employs a single path to connect its base link to its terminal link. In this configuration, proximal links (links closer to the base) in a chain are required to carry and move the more distal links (those closer to the end point) and their actuators, resulting in increased linkage inertia and decreased linkage stiffness.

Conversely, a parallel mechanism joins its base and terminal links across multiple pathways. In the case of our joystick, its links are arranged in a fully parallel mechanism that allows for the attachment of actuators to the ground, significantly reducing the system inertia. Moreover, because a
Parallel mechanism uses multiple pathways arranged in one or more closed kinematic loops, the resulting device can be made to have higher stiffness than serial mechanisms, which in turn allows for high bandwidth force transmission. Other haptic devices that make use of a parallel configuration include Gough-Stewart platform-like haptic devices [5-7], wire-actuated [8-10], and spherical mechanisms [11-13] similar to the scheme chosen for our design.

For several years, UC Berkeley and NASA Ames have collaborated on the development of joystick-type haptic devices based on a common underlying parallel planar-spherical spatial kinematic architecture [1]. The haptic devices described in this paper represents the most recent iterations of this research. The first joystick produced used a large set of links and provided for full arm-scale motion in the joystick workspace [12]. This device was later scaled down to provide a workspace compatible with the smaller finger and hand motions associated with using a writing instrument or a computer mouse [13]. The current joystick maintains the finger-motion-sized workspace of the previous device and improves both the joystick linkage design and the control hardware and software.

To achieve realistic haptic simulation, a tightly regulated, temporally deterministic response from the haptic interface is needed. We originally selected a Digital Signal Processor (DSP) based architecture allowing us to isolate time-critical haptic interface control tasks from computationally intensive and more slowly updated Virtual Environment (VE) simulation processes. The standalone DSP is embedded inside the haptic interface itself, maintaining portability by allowing the device to be connected with different VE simulation computers. A Field Programmable Gate Array (FPGA) was included to provide the high-speed dedicated I/O for reading haptic interface sensors and commanding actuators.

While the DSP/FPGA based design met the performance requirements, the custom control hardware was both expensive to produce and did not provide the robustness necessary for regular experimental use. A second generation of control hardware was designed using only commercially available off-the-shelf components, allowing it to be assembled in prototype quantities quickly and inexpensively. This version is based on an embedded x86 microprocessor and a commercially available motion control processor. This new control electronics package maintains the portable self-contained overall design of the haptic device.

These joystick devices have been used in haptic demonstrations in our labs, including investigations of texture perception [12,14]. The new metal joystick described in this paper is intended for ongoing human performance research at NASA Ames.

**DEVICE OVERVIEW**

All versions of this haptic device share a common joystick linkage architecture (Fig. 2) that contains 10 rigid links with 12 revolute joints, in a parallel spherical mechanism arranged in three closed kinematic loops. The device provides point-force interaction over a three-dimensional translational work volume and has evolved from previous joysticks designed in our lab [1]. The joystick is manipulated through various stylus that attach to a magnetic ball joint on the endpoint. The links are passively mass balanced so gravity forces cannot be sensed by the user as the end effector is moved.

Actively controlled force is provided by three brushless DC motors with housings fixed to the joystick enclosure so the linkage does not carry their weight. Open loop force and motion transfer fidelity is improved due to the lack of transmission elements such as gears or cable drives that would otherwise increase friction and add backlash. Sinusoidally wound, brushless DC motors are used, providing up to 5N of force at the joystick endpoint for short durations (2 sec, based on amplifier and motor specifications). Because sinusoidal motor commutation is matched to the motor winding profile, torque ripple—spatially dependent periodic deviations from commanded actuator torque—is eliminated, providing smooth feeling surfaces during joystick simulations. The use of analog sinusoidal encoders and sinusoidal interpolation provides high-resolution angular position feedback from the motors.

Software has been written to provide communication between the joystick and an external personal computer (PC) through the following interfaces: a standard PC serial port, Universal Serial Bus (USB) ver.1.1, and a standard PC parallel port interface. This communication allows the joystick endpoint position and velocity data to be sent from the joystick to the PC to continuously update a virtual environment and to elicit a force command from the PC virtual environment to the joystick, where it can be felt by the user. The parallel port provided best performance for haptic communication. Testing of the parallel port link between a PC and the second generation x86 based control electronics has confirmed a maximum frame rate of 1.5 kHz (1500 packets of information per second). Since each packet includes 700 bits of data, the data rate to PC is approximately 1 Mbps.
SPACIAL LINKAGE

For a full derivation of the joystick kinematics, the reader should refer to Adelstein, Ho and Kazerooni [1]. The joystick mechanism consists of two spherical loops and one planar loop. The first loop is composed of links 1, 2, 3, 4 and 5 and the second contains links 5, 6, 7, 8 and 1. (See Fig. 2 for the full linkage assembly and link labeling.) These two loops form a spherical linkage because the axes of all its joints intersect at one common point. The final loop comprising links 4, 5, 6, 10, and 9 form a planar parallelogram structure.

The shape of the links has changed significantly from our previous implementations so as to attain the mass balance and workspace improvements discussed later in this paper, but the relative locations of the revolute joints have not changed. The new linkage is kinematically equivalent to the simplified sketch of the original linkage architecture shown in Fig. 2. One of the most pronounced changes is that the new linkage (Fig. 5) has the end effector inverted such that it is directed in the positive z direction, thereby increasing the usable workspace. Other less visible changes were made to the relative orientation of several links within the spherical portion of the mechanism.

The coordinate system used to describe the linkage kinematics is shown in Fig. 3. The origin is located at the intersection of the three motor shafts. The positive x-axis points into the \( \Gamma \) motor shaft, the positive y-axis points out of the plane of the motor shafts, and the positive z-axis points into the \( A \) motor shaft. Positive angle measurements are defined according to the right hand rule about axes pointing from the origin to motors \( A \), \( B \), and \( \Gamma \). Zero angle measurements are defined as the x-z plane containing the motor shafts.

The derivation of the forward kinematics of the linkage that map the joint angles of the actuators to the Cartesian coordinates of the end effector and the system Jacobian that relates the endpoint force to the actuator torque can be found in [1].

PASSIVE MASS BALANCE

Balancing of the linkage in haptic devices is a technique used extensively in robotics. When implemented, the user does not need to provide a supporting force, imparting a lighter feel to virtual environment simulations [15]. By balancing the linkage passively (as opposed to an active balance that relies on power from the actuators [16]), we are able to reduce computational burden for the embedded processor. The passive balance also allows for a smooth simulation of a zero-force condition in the virtual environment because the actuators can be completely unpowered.

Passive mass balance of the linkages is achieved if the torque induced by gravity, \( \tau_i \), for each actuator \( (i = 1, \ldots, 3) \) remain at zero. This torque can be defined in terms of the partial derivative of the gravity potential energy of the system with respect to each actuator coordinate. In equation form,

\[
\tau_i = \frac{\partial U}{\partial q_i} = 0
\]  

where \( U \) is the potential energy of the whole system and \( q_i \) is the \( i^{th} \) system coordinate. Equation (1) will be satisfied if the derivatives of \( U \) are forced to be constant for all \( q_i \). The equations for the derivatives of the potential energy, \( U \), are expressed in terms of the mass and location of the mass center for each link and its attached counterweights in the local link coordinate frame and the displacement of the link mass centers above a datum as a function of actuator angles. The mass and center of mass location are specific to the designed set of
Figure 4: The coordinate system used to define the link dimensions.
In this example, \( D_b \) is greater than zero and for base links 2, 5, and 8 the origin frame \( (x^*_a, z^*_a) \) is at the mechanism center. Link pairs corresponding to link \( a \) and link \( b \) in this diagram are 3 and 2, 4 and 5, 6 and 5, and 7 and 8, respectively. For all planar links, \( L_b \) is equal to zero.

joystick links. The actuator angles are not dependent on the shapes of the links in the joystick design (given that all designs are kinematically equivalent), but rather on the instantaneous position of the endpoint. The various terms dependent on link mass and center of mass are grouped according to the specific trigonometric functions of actuator angles terms that they multiply. These groupings of link properties become coefficients, which when scaled by gravity acceleration, \( g \), multiply terms that are dependent solely on actuator angle. If each of these coefficients can be driven to zero, the derivatives of the potential energy would be driven to zero, and the potential energy, \( U \), would also be constant. The system would thereafter satisfy the passive mass balance condition expressed by the partial derivative condition of Eq. (1).

The potential energy derivatives, regardless of linkage pose, are defined when the fixed base is aligned perpendicular to the gravity vector, according to the following three equations:

\[
\frac{\partial U}{\partial \alpha} = -k_{2\gamma} s\alpha + k_{6\gamma} c\alpha - k_{7\gamma} \frac{s^3 \beta}{d^3} s\alpha \ldots
\]
\[
+ k_{8\gamma} \frac{c\alpha c^2 \beta}{d^3} - k_{9\gamma} s\alpha c\alpha s\beta \frac{c^2 \beta + Q_{\beta \alpha}^2}{d^3} \ldots
\]
\[
- k_{10\gamma} \frac{c^3 \gamma s\alpha}{d^3} + k_{11\gamma} \frac{c\alpha c^2 \gamma}{d^3} \ldots
\]
\[
- k_{12\gamma} s\alpha c\alpha s\gamma \frac{c^2 \gamma + Q_{\gamma \alpha}^2}{d^3}
\]

(2)

(3)

(4)

\[
\frac{\partial U}{\partial \beta} = k_{4\gamma} c\beta - k_{7\gamma} \frac{s^3 \beta^3 c^3}{d^3}
\]

(5)

(6)

Where \( d_\beta = \sqrt{1 - s^2 \alpha s^2 \beta} \)

\( d_\gamma = \sqrt{1 - s^2 \alpha s^2 \gamma} \)

The coefficients used in Eq. (2)–(4) are expanded in Table 1. They are subscripted \( y \) to acknowledge the vertical gravity vector as being in the \( y \) direction.

Table 1: Expanded coefficients of the potential energy

| \( k_{1\gamma} \) | \( m_2g y^2 \) |
| \( k_{2\gamma} \) | \( m_3g y_3 \) |
| \( k_{3\gamma} \) | \( m_4g y_4 \) |
| \( k_{4\gamma} \) | \( m_2g x_2 + m_3g (D_1 + H_2 + z_3) \) |
| \( k_{5\gamma} \) | \( m_7g (D_2 + H_8 + z_7) + m_8g x_8 \) |
| \( k_{6\gamma} \) | \( m_4g (D_1 + H_2 + z_4) + m_3g x_3 + m_6g (D_6 + H_5 + z_5) + m_5g x_5 + m_10g (D_{10} + D_4 + D_9 + H_5 + z_{10}) \) |
| \( k_{7\gamma} \) | \( m_3g x_3 + m_4g x_4 + m_9g H_{10} + m_{10g} (D_{10} + H_{10}) \) |
| \( k_{8\gamma} \) | \( m_3g y_3 \) |
| \( k_{9\gamma} \) | \( -m_4g y_4 + m_{10g} y_{10} \) |
| \( k_{10\gamma} \) | \( -m_6g x_6 - m_7g x_7 + m_9g x_9 + m_{10g} H_{9} \) |
| \( k_{11\gamma} \) | \( -m_7g y_7 \) |
| \( k_{12\gamma} \) | \( m_6g y_6 - m_9g y_9 \) |

In Table 1, \( m_i \) is the mass of the \( i^{th} \) link; \( g \) is the gravity acceleration; \( x_i, y_i, \) and \( z_i \) are the coordinates of the center of mass of the \( i^{th} \) link measured from the local link coordinate system; and the remaining variables are link dimensions. The link dimension coordinate systems are defined according to Fig. 4.

Seven of the coefficients defined in table 1 are immediately reduced to zero because of the trivial symmetry that occurs when individual link centers of mass are located in the plane of the \( x-z \) axes of Fig. 4, i.e., where \( y_i = 0 \). This leaves only coefficients \( k_{4\gamma}, k_{5\gamma}, k_{6\gamma}, k_{7\gamma}, \) and \( k_{10\gamma} \). Furthermore, a solution to these equations can be shown to guarantee mass balance for all gravity vector orientations even though Eq (2)-(4) are derived for the case where the linkage base is horizontal.

The values of each coefficient were calculated in a spreadsheet as link mass and dimensions and counterweight mass and placement were manually varied in an iterative process to produce the final static balance. In addition, holes
were bored through the non-counterweight portions of links to decrease the amount of counterweight mass needed and to reduce the overall system inertia. Tests with standard masses confirmed that the maximum vertical force required to support the unpowered linkage for the final designs was less than 0.02 N (2 g) for all endpoint locations tested within the workspace.

With repeated design iterations, the residual force associated with any remaining mass imbalance is negligible in comparison to the bearing friction and other imperfections associated with joint alignment and linkage dimension precision in the actual system. The friction and alignment related forces in the final joystick were insignificant enough that a test user did not feel that her motions were impeded when the machine was unpowered. The mass balance design was considered complete when the unpowered linkage could be released at any position inside the workspace with no resulting movement of the end effector.

The changes that occur as link dimensions were modified to improve mass balance can introduce collisions between links that would decrease the joystick’s useable, interference-free workspace. Therefore, as modifications were made, workspace extent was monitored prior to fabrication on the CAD parts using the visualNastran® kinematic simulation package from MSC Corp. The formulation of the mass balance problem is well suited to a numerical optimization approach. However, at the time this research was conducted there was not an easy way to incorporate a workspace analysis into an optimization penalty function and the manual technique described above was expedient and successful. The numerical optimization approach presents an interesting avenue for future research with this hardware. The final result of the mass balance and link redesign provided a larger workspace than did previous versions of the small joystick.

**LINKAGE MANUFACTURE**

The first linkage (Fig. 5-A) was constructed from polycarbonate plastic parts on a Fused Deposition Modeling (FDM) rapid prototyping system (Stratasys Inc., Model: TITAN) in the laboratory at UC Berkeley. The FDM process allowed for the rapid construction and testing of several prototypes of the linkage. Brass extrusions were chosen as the counterweight material for their density, low cost, and ease of machining. The penalty for using the FDM fabrication process was greater linkage compliance due to the FDM plastic, poorer dimensional tolerance due to the 0.25 mm precision of the FDM machine, and brittleness which made the sections of the links near the bearings and counterweights prone to crack during assembly.

In parallel, another version of the linkage (Fig. 5-B) was designed using a combination of 6061-T6 alloy aluminum and 304 alloy stainless steel links. The all-metal construction enabled tolerances significantly better than the 0.25mm of the FDM process. The precision and stiffness of this design make it more suitable for use in the human performance experiments. However, the higher manufacturing cost makes the design less appealing commercially.

Various graspable stylus designs were also created and can be connected to the linkage via a low friction magnetic ball joint that is coupled to the endpoint (at Link 10) of the joystick.

**SINUSOIDAL ENCODER INTERPOLATION**

Incremental optical encoders measure shaft rotation angles by detecting the motion of a uniform radial grating pattern of alternating transparent and opaque lines arranged around the circumference of disk fixed to the shaft [17]. Typically, a threshold is applied to the output of the sensor detecting the varying light pattern to give a square-wave (high/low) output from the encoder. Encoders usually have two tracks of grating patterns (typically labeled A and B) offset circumferentially from each other by one-quarter of the grating period such that there is a 90° spatial (termed quadrature) phase shift between the two output traces. With two tracks and two possible states per track (high/low), a total of 2 bits of information is available, giving a fourfold increase in encoder resolution over the disk’s grating line density. This resolution enhancement and subsequent angle accumulation (i.e., counting of state transitions) is typically carried out by integrated electronic quadrature decoder/counter circuits.

To achieve angular resolution levels well beyond those achievable through quadrature decoding, we use sinusoidal encoders and implement a sinusoidal interpolation scheme. Sinusoidal encoders’ mechano-optical components operate like those described above but provide the individual 90° phase-shifted quadrature channels as differentially amplified,
continuously-varying analog sinusoid signals, without conversion into two-state digital logic levels.

To compute the interpolation, the continuous channel A and B signals first undergo analog-to-digital (A/D) conversion. A/D conversion of the sinusoids results in digitized signals that are discretized into uniform amplitude increments as depicted schematically in Fig. 6 for operation about a single encoder grating line. Interpolation of the phase angle for each sinusoid — i.e., \( \alpha = \arcsin(A) \) and \( \beta = \arcsin(B) \) — requires calculation of the inverse sine of each signal. However, because the inverse sine function is a nonlinear mapping, the uniform A/D conversion (ADC) increments yield variable size angle increments. Consequently, interpolated phase angle increments plotted in Fig. 6 show highest density sampling as the sinusoids’ amplitudes approach zero from either above or below.

From the overlapping sinusoidal curves in Fig. 6, it is seen that the higher phase angle sample densities span 90° segments that alternate between the two encoder channels A and B. These 90° segments correspond to quarter rotations through individual grating lines equivalent to the conventional encoder quadrature resolution. Thus, by switching every 90° between the two curves whenever they pass through \( \pm 0.707 \) of the peak amplitude, it is possible to maximize the effective angle interpolation resolution across each encoder grating cycle by using only the most densely sampled regions of each inverse sine function.

Because there are a fixed number of discrete sinusoid ADC levels, it is more efficient in practice to retrieve arcsine values from a pre-computed lookup table (LUT) than it is to repeatedly compute on-line floating-point calculations. Additionally, encoding LUT angle entries as fixed-point binary numbers enables the subsequently described interpolation steps to be carried out with simple Boolean operations, further avoiding resource-intensive floating point math. Finally, limiting the size of the LUT helps reduce the amount of memory required for implementing the interpolation scheme.

From Fig. 6, it is noted that the alternating 90° high resolution regions for each channel (A or B) are composed of successive mirrored 45° segments: the first from the sinusoid’s negative amplitude portion prior to the zero-crossing, and the second from the positive portion immediately following the zero crossing. Hence, by taking advantage of the signal’s sign, the inverse sine LUT theoretically needs only enough storage to correspond with 45° (1/8) of the sine wave.

**INTERPOLATION SCHEME INTERPRETATION**

The encoders used in both versions of the device discussed in this paper have incremental 1024 line sinusoidal output (Computer Optical Inc.; Model CP-5150). In the first generation device, 8-bit differential A/D converters sample encoder channel A and B voltages, with the ADC output values spanning integers \(-128\) to \(+127\) for the resultant \(\pm 1\) volt sinusoidal signal inputs. The quadrant of the current encoder position is simply determined from the sign of the ADC value of channel A and B (i.e., by checking the ADC most significant bit), as shown in Fig. 7. As noted above, the signs and therefore the quadrants correspond to the standard encoder quadrature signals.

Within each quadrant, the interpolation input alternates every 45° between the higher resolution segments of channel A and B. For this reason, the LUT only contains values corresponding to positive ADC values between 0 and 90. As seen in Fig. 7, the selection between channels A and B is based on which has the lesser absolute ADC magnitude. Combining the A/B selection with the quadrant information allows computation of the interpolated phase angle between 0° and 360° for the individual encoder line. Thus, the interpolation is composed from 2 bits for the four quadrature quadrants of the 360° cycle, 1 bit for which half of the quadrant is being
interpolated, and a 7-bit LUT density for the 45° segment, for a
total of 1024 counts per single encoder grating line. With 1024
lines and 1024 counts per line, the 1024-line encoder produces
1,048,576 counts per revolution. However, as discussed above,
some arcsine LUT locations are not addressable by possible ADC
levels, resulting in local fluctuations of the interpolation
resolution. This degrades the effective resolution to 524,288
counts per revolution, or 0.00069°. At a nominal 10 cm (4 inch)
radius for the endpoint of our hand-scale haptic interface, the
worst-case sensor resolution corresponds to a theoretical
displacement of 1.2µm (4.7 × 10⁻⁵ inch), ignoring any possible
linkage mechanism compliance, backlash, etc.

INTERPOLATION SCHEME PERFORMANCE

The A/D converters used in our design while rated to 20
MSPS (20 million samples per second), are run at 1 MSPS
synchronously with FPGA based quadrature and interpolation
circuitry. At 1 MSPS, the ADC and quadrature counters are
theoretically capable of capturing all up-down transitions
(4 samples/line × 1024 lines) for encoder speeds up to
244.2 rev/s (14.6 KRPM). However, since encoder quadrature
in our case is derived from the sign of the sinusoidal signals,
the coarse resolution (4 × 1024 counts per revolution) depends
solely on each ADC levels’ MSB (most significant bit).

Because the interpolation also operates at the 1 MSPS
FPGA rate and because the interpolator output is written out to
the data bus as a parallel word, full 19-bit high-resolution can
be achieved up to the same 14.6 KRPM rotation speeds as the
low resolution quadrature count. This performance differs
significantly from that expected of the typical 10 to 20X
commercially available interpolated encoders. Following the
interpolation stage in these commercial sensors, the n-bit wide
parallel output is converted back into a dual-track square-wave
output so that the resulting quadrature signals can then be
decoded and counted by the same interface boards and circuits
employed for conventional low resolution (e.g. 1024 line)
encoders. Consequently, these commercial devices are limited
to operation at slow velocities in order to prevent intervening
interpolation level transitions from occurring too quickly to
contribute to the pulse pattern. If transitions between adjacent
levels are not detected, the pulse train will fail to keep up with
shaft rotation and the external quadrature counter circuits will
be inaccurate due to the slippage. For example, if our high-
resolution output were to be converted back to a quadrature
pattern at the full 1 MSPS FPGA rate, counts would be lost if
the shaft speed were to exceed 114 RPM (i.e., 1.9 rev/s =
1 MSPS / 524,288 counts/rev).

CUSTOM FPGA BASED INTERPOLATION

HARDWARE

We confirmed that quadrature-level resolution could be
obtained from the encoder FPGA circuitry at rates far
exceeding the minimum 1 kHz target for haptic interface
control. The interpolated angle from the FPGA output was
displayed via a D/A converter on an oscilloscope. Approximately uniform input velocity produced by connecting
a small DC motor to the encoder resulted in the expected
sawtooth angle trace (ramp-up until digital value roll-over) that
could then be compared with the once-per-revolution encoder
index pulse. No slippage between the index pulse and the ramp
waveform was noted after several minutes at motor speeds of
150 rev/s (9000 RPM), indicating that the ADC quadrature
update, and therefore the high-resolution interpolation output,
could be maintained up to rates of 614.4 kHz (150 rev/s ×
1024 lines/rev × 4 counts/line).

Experimental validation is currently underway to assess the
 interpolator’s spatial fidelity. Particular interest is given to each
grating line’s sinusoidal cycle, the interpolations of which are
subject to inaccuracies (systematic errors) caused by the
sensor’s internal electro-optical properties, sensor misalignment
at the time of assembly, and imprecise electronic adjustment.
These factors can produce nonzero offsets and unmatched or
improperly scaled gains for channel A and B encoder output
signals, as well as imperfect (non-90°) phasing between the
channels. Additionally, the encoder’s signals can deviate
significantly from the ideal sinusoidal profile [18]. Moreover,
the signals’ waveform, gain, and offset may also vary from line
to line around the circumference of the encoder disk.

Initial observations for our haptic interface’s three
encoders indicate that signal profiles very closely resemble
sinusoids, especially within the 45° segments employed by our
interpolation. A preliminary analysis of interpolator fidelity
was conducted by examining the deviation of the local angle
from the expected best fit to the constant velocity displacement.
Systematic departures from the best-fit angle were 0.0027°,
0.0041°, and 0.0027° RMS (0.008°, 0.010°, and 0.0045° at
worst case peaks) across individual grating lines for the haptic
interface’s three encoders—prior to any adjustment of LUT
output, or external trimming of raw encoder offsets, gains, and
phase. Stochastic variability in interpolation output due to
sensor or other system noise typically amounted to ±0.0007°
(1/500,000 rev) or less when the interface’s servo amplifiers
and actuators were not powered.

COTS BASED INTERPOLATION HARDWARE

In the second generation device, commercially available
encoder interpolation modules were substituted for the custom
hardware and software described above in the interest of
lowering the cost and increasing the robustness of the system.
A standalone encoder interpolator module (Computer Optical
Inc.; Model CP-1064-16-1) was added between each encoder
and the embedded control computer. The module performs 16X
sine interpolation and outputs a standard quadrature signal.
With this sine interpolation, and 4X quadrature decoding, the
angular position resolution information is 1024x16x4 ∼ 65.5K
counts/rev. At the joystick endpoint’s maximum reach of 10
cm, this corresponds to a theoretical resolution in x-y-z
endpoint space of 9.6µm. The actual resolution of the endpoint
position is coarser due to factors such as compliance of the
links and joints.

MOTORS AND COMMUTATION

Both designs share the same brushless DC motors (BLDC)
for actuation (Pittman Motors; ELCOM Model 4141). The
motors have four-pole, three-phase slotless sinusoidal
windings. The absence of brush-rotor contact in BLDC drives eliminates a source of shaft friction and is therefore beneficial to the “feel” of haptic interface hardware. Most importantly, we employ BLDC motors in our haptic devices because of their higher continuous and peak torque output per unit of actuator size or mass.

The motors were chosen with sinusoidal windings because they provide a constant torque profile when commutated sinusoidally, thus eliminating torque ripple which would be transmitted to the user and would degrade the haptic simulation. At its rated continuous torque, the motor produces approximately 1.4 N (measured) at the end effector when the linkage is fully extended to its maximum 10 cm reach. This force level is sufficient to cover the majority of precision finger-scale manipulation tasks [3]. In addition, a peak force of 5 N can be applied for durations less than 2 seconds (duration limited by the amplifiers).

Commercial pulse width modulated (PWM) servo amplifiers were selected for the haptic interface because of their higher power efficiency and consequent lower heat losses than linear amplifiers. The compact sinusoidal PWM amplifiers (Advanced Motion Controls; Model S16A8) provide a maximum 8A of current per motor (or 75 W per motor based on the motor winding resistance). The PWM amplifiers switching frequency is 33 kHz, well beyond the 20 kHz human audible range and beyond the mechanical bandwidth of the system. The three motors and amplifiers are contained within the desktop joystick housing. The top and bottom plate, onto which the motors and amplifiers are respectively mounted, also serve as heat sinks.

Proper commutation of motor windings, i.e., the energizing of winding phases in the correct sequence, depends on knowledge of the instantaneous angular displacement between rotor and stator. Brushed drives, by design, ensure that the simple mechanical “switching” of the brushes between the rotating commutator bars will always activate the correct phases with respect to the permanent magnet field, without need for additional sensor feedback. In BLDC drives, there is no such direct mechanical connection to account for the angular rotation between rotor magnets and stator windings. Thus, trapezoidal and sinusoidal commutation, the most widely used commutation schemes for BLDC machines, depend on motor shaft angle sensors for this information.

TRAPEZOIDAL COMMUTATION

In trapezoidal commutation, Hall Effect sensors are used to detect the orientation of the rotor magnetic field. Typically, three Hall Effect sensors are distributed around the stator circumference to match the 120° electrical offset between the three winding phases. The two-state signal transitions from the three Hall sensors serve to divide the rotation into unique 60° wide segments for each motor magnetic pole pair. Switching logic under direct control of the Hall Effect sensor states will direct current among the trapezoidal amplifier’s three output channels to produce the six-step four-level delta-connected winding current profiles depicted in Fig. 8. This straightforward current switching strategy makes trapezoidal (sometimes called “six-step”, or “brushless DC”) servo amplifiers and sensors simpler, less expensive, and, therefore, the more popular of the brushless commutator techniques.

Fig. 8 Torque output for 6-step Hall Effect sensor-based trapezoidal commutation of sinusoidally wound delta-connected brushless DC motor. (Upper left) Hall Effect sensor output; (middle left), three level brushless DC servo amplifier channel input currents to delta-wound motor nodes; (lower left), resultant four level delta winding currents superimposed over underlying sinusoidal winding current-to-torque (back EMF-to-speed) profile; (right) normalized individual phase torques and their sum.
SINUSOIDAL COMMUTATION

From Fig. 8, we see that the torque ripple arising from trapezoidal commutation coincides with significant departures in winding currents from the underlying sinusoidal current-to-torque (or back EMF) profile. Essentially, the BLDC motor ripple is caused by the four-level switched waveform’s overly coarse sinusoid approximation, and can therefore be remedied by refining the commutation profile.

Ideally, by matching individual phase currents to the particular motor winding’s spatial and temporal torque characteristics, sinusoidal commutation can completely eliminate torque ripple as shown in Fig. 9. However, as noted from Figs. 8 and 9, a perfectly matched sinusoidal (also called “AC brushless”) amplifier will reduce mean torque to ~91% (i.e., 1.5 / 1.65) of that obtainable with trapezoidal commutation for a given sinusoidally wound BLDC motor.

Depending on the particular servo amplifier design, either encoders with direct angular output or resolvers that automatically convert angle to commutation-ready sinusoidal encoders with direct angular output or resolvers that incorporate the modulation of the desired torque command by the commutation profile, and produces three-channel PWM current output. Two of the amplifier output channels directly follow the input control phases. The amplifier derives the third phase internally under the assumption that the three channels’ instantaneous currents sum to zero. Because the low-level control is software configurable, six-step or other balanced non-sinusoidal commutation patterns may also be implemented on the amplifiers.

IMPLEMENTATION OF CUSTOM FPGA BASED COMMUTATION

The modulation of the commanded input torque by the respective windings’ commutation profile entails real-time multiplication on the FPGA of the torque command by the appropriate angle-dependent sinusoidal levels. The commutation profile, pre-computed for equally spaced angle intervals, is stored on the FPGA as a single 256-point sine curve LUT. Though the LUT only covers the first quarter commutation cycle (0° to 90°), the same sine values can also be used for the remainder of the cycle—either directly (from 90° to 180°) or with a sign inversion (180° to 360°). Moreover, to span the entire 10-bit commutation cycle (210 = 4 × 256 points per segment), only the 11 most significant (two electrical cycles per mechanical revolution) bits of the interpolated encoder reading are required to address LUT angle entries.

The modulation process proceeds in the following order. First, prior to addressing the sine LUT, the encoder angles are adjusted for offsets between the initializing index pulse and the individual 120°-shifted winding phase orientations. Second, sine LUT output values for the offset-adjusted angles are then multiplied by the torque command. Finally, the two 120° shifted control signal channels resulting from the multiplication (i.e., the commutation-modulated torque commands) are issued via a pair of bipolar 14-bit digital-to-analog converters (DACs) to each servo amplifier. Since there is only one LUT for this process that is shared by the haptic interface’s three motors, the sinusoidal profile cannot be applied simultaneously to each of
the amplifier input channels. Instead, there is a 2 µs time skewing between the loading of successive DAC channels, contributing to a cumulative six channel latency of 12 µs (6 × 2 µs). The individual DAC’s 16 µs settling time, which limits the maximum commutation update rate, but is long enough that the commutation cycle is not further slowed by the accumulated inter-channel delay. The resulting 62 kHz FPGA commutation update rate is still nearly double the commercial servo amplifiers’ nominal 33 kHz PWM switching frequency.

Subjectively, we observed that the implementation of sinusoidally commutation in our hand-scale haptic device has eliminated any perceptible evidence of ripple, which had been significant under the prior trapezoidal commutation at 30⁰ intervals. We have not yet attempted formal calibration of the new motor and consequent overall haptic interface dynamics under the improvements afforded by the sinusoidal drives.

DSP/FPGA BASED CONTROL HARDWARE

The DSP/FPGA based control electronics used an embedded DSP based controller (Innovative Integration, Model: SBC6x) to perform what we refer to as mid-level functions: kinematic transformations, closed-loop servo-motor control, workspace boundary enforcement, and active mass counterbalance. In addition, the embedded controller managed either USB or serial communication with a PC running a virtual environment (we refer to this as high-level control). Low-level control, which included encoder interpolation, velocity derivation through a digital filter, and servo-motor commutation, was performed by a custom-designed add-on board that attaches to the embedded DSP controller. The add-on board contained the FPGA for performing these functions as well as analog-to-digital converters (ADC), digital-to-analog signal conditioning hardware for sampling encoders and outputting motor commands. The FPGA allowed many functions to be integrated into a single chip and, because the FPGA can be reprogrammed, provided a flexible platform for experimentation and debugging. While this custom-hardware control system for the joystick performed adequately in the original prototype, it was both difficult and expensive to duplicate in the laboratory due to noise and signal integrity issues on the custom FPGA add-on board and the fact that the all surface-mount component design made in-house assembly impractical. While we feel the FPGA based architecture would be a logical choice for a production version of the joystick, the decision was made to switch to a COTS control hardware architecture to reduce the cost for future prototypes and create a more rugged/reliable experimentation platform for use in the lab.

COTS CONTROL HARDWARE

The control architecture for the latest generation haptic joystick shares the same low-mid-high level control division as the DSP/FPGA version, however we have chosen different approaches for each to make an all off-the-shelf system and reduce cost significantly. Mid-level control is now managed by an embedded x86 based microprocessor and the combination of a dedicated motion control chip and an encoder interpolation module take care of low-level functions. Together, they form the stand-alone controller that is housed inside the joystick base enclosure along with the motors and servo amplifiers.

The mid-level control is performed by the embedded Intel 386EX microprocessor on a development board from Tern Inc. (Model: i386-Engine). The mid-level board is responsible for the servo control function that calculates the commanded interaction forces as well as the workspace boundary
enforcement function that prevents motion into regions where
link interference would occur. The mid-level control also
includes a parallel communication interface to an optional
external computer that can run virtual environment simulations.

The low level control is carried out on the Term MotionC-S
development board with PMD Navigator® Series motion
processor. The motion processor performs quadrature decoding
to give angular position for each axis. It also estimates velocity
by keeping track of the number of encoder counts over a known
time window. The low-level embedded controller can vary the
size of this window to improve the velocity estimate for slow
motions. The PMD chip is also responsible for the sinusoidal
motor commutation.

SOFTWARE
The joystick appears to the user as an impedance-
controlled device. Under impedance control, the input is the
motion of the endpoint generated by the human. This motion is
sensed by the device’s encoders and transformed to motion of a
virtual endpoint in a virtual environment running on the either
on the embedded processor or an external personal computer.
The output signal (feedback) to the user is the point force
generated by the joystick at its end effector. Our device is
intended to interface with existing virtual environment software
for the PC. This virtual environment software determines what
force should be generated at the end effector based on its
location in the virtual environment. For example, if the end
effector in the virtual environment were to strike a virtual
“wall,” a force proportional to the depth of penetration of the
virtual wall could be generated. We also developed a simple
API for the haptic device controller to interface with the PC
virtual environment software. The API includes functions to
obtain the end effector position and velocity, and can output a
three-component force vector at end effector force. This way
the virtual environment developer can utilize the API to create
new Microsoft Windows applications that incorporate the
relationship that best fits the experiment or haptic environment
he or she is trying to create.

The software architecture is shown in Fig. 10. The local
(mid-level) processor converts the encoder counts for each
actuator axis into angle measurements. The forward kinematic
relations are then applied to obtain the end effector position and
velocity in Cartesian coordinates. The state of the end effector
is made available to the external PC running the virtual
environment through the API. Based on this information, the
external PC determines the appropriate force in Cartesian
coordinates to apply at the end effector. It then sends this out to
the local processor through the API, and the user is given the
option to command the forces directly (and close the loop on
the external PC) or use a PID based impedance control running
on the embedded processor. In either case, the force command
is converted into actuator torques using the instantaneous
Jacobian and sent out to the actuators. The local processor also
performs workspace boundary enforcement in the actuator
angle coordinate space by placing virtual haptic “walls” near
the boundaries of the physical workspace to prevent the user
from colliding the linkage into itself or the device housing.

Convincing haptic simulations are widely reported to
require update rates in excess of 1 kHz [19]. The local haptic
environment running on the embedded microprocessor can be
updated at a rate of up to 1.5 kHz. If other haptic calculations
are occurring on the PC, the endpoint position and velocity and
force command data must be transferred at 1 kHz between the
PC and local microprocessor.

The DSP/FPGA version of the haptic joystick relied on
either an RS232 serial or USB ver.1.1 communication link with
the VE. The serial link was limited to the maximum PC serial
port speed of 115.2 kbps. While USB has become a standard
high-speed interface for PCs and seems like a promising
technology for haptic communication, it does so by transmitting
large blocks of data at a fairly slow rate: a maximum of 500 Hz.
for USB 1.1 [20]. The new USB 2.0 standard, with a frame rate
eight times higher than USB 1.1, was not available at the time
our system was developed. In the COTS control hardware
version of the joystick, a standard high speed PC parallel port is
used and a 1 kHz frame rate, 1 Mbps data transfer rate
connection was established between the PC and the haptic
device. While sufficient for a convincing haptic simulation, the
data rate is intentionally made less than the maximum achieved
in testing to guarantee performance in the Windows PC
environment.

CONCLUSIONS
We have designed a three-degree-of-freedom force-
feedback haptic joystick with the goal of combining portability
and compact size with the performance, spatial resolution,
continuous force magnitude, smoothness, balance, and haptic
update rate necessary for precision finger-based interaction.
The current iteration of the joystick uses links kinematically
equivalent to past versions. However, the new design provides
a larger workspace and is passively mass balanced.

Two versions of the mechanical linkage were prototyped: a
low cost FDM polycarbonate linkage potentially suitable for
mass production and a precision all-metal linkage designed for
haptics research. However, because of dimensional imprecision
and excessive compliance, further development of the FDM
version has been postponed for now. Sinusoidal encoder
interpolation and sinusoidal motor commutation have been
implemented to achieve high resolution, smooth haptic
simulation.

Software has been written to allow for high-speed data
transfers between the joystick and the PC, and a simple API is
also provided to allow the haptic device to interact with other
virtual environment software. The DSP/FPGA based control
electronics design has been completed and tested. The function
of individual device subsystems for the COTS based control
electronics design have thus far been tested separately. For this
latest generation of the device, we expect shortly to begin the
process of integrating the communication software and API
with the low-level control code on the embedded platform and
connect the control system to the actuated linkage. Testing of
the completed hardware and software system for the COTS
system will follow.
ACKNOWLEDGEMENTS

The authors would like to thank Dr. Bernard D. Adelstein for his support and guidance on this project. Work at UC Berkeley was supported by NASA-ARC Cooperative Agreement NCC2-1347, NCC2-1171, and NASA RTOP 704-10-82 (IT Base). Additional support was provided through the NASA Intelligent Systems Human Centered Computing Program, and a NASA-Ames Director’s Discretionary Fund Award.

NOMENCLATURE

Abbreviations:

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application Program Interface</td>
</tr>
<tr>
<td>BLDC</td>
<td>Brushless DC (type of electric motor)</td>
</tr>
<tr>
<td>CAD</td>
<td>Computer Aided Design</td>
</tr>
<tr>
<td>COTS</td>
<td>Commercially available Off-The-Shelf components</td>
</tr>
<tr>
<td>dpi</td>
<td>Dots per inch</td>
</tr>
<tr>
<td>FDM</td>
<td>Fused Deposition Modeling</td>
</tr>
<tr>
<td>PC</td>
<td>Personal Computer</td>
</tr>
<tr>
<td>PID</td>
<td>Controller with output proportional to the error, the error integral, and the error derivative</td>
</tr>
<tr>
<td>TTL</td>
<td>Transistor-Transistor Logic (digital signal standard)</td>
</tr>
<tr>
<td>USB</td>
<td>Universal Serial Bus</td>
</tr>
<tr>
<td>VE</td>
<td>Virtual Environment</td>
</tr>
<tr>
<td>kbps</td>
<td>Thousand bits per second (data transfer rate)</td>
</tr>
<tr>
<td>x86</td>
<td>Generic name for a series of Intel processors</td>
</tr>
</tbody>
</table>

Variables Used in Paper:

- $D_b$: Link thickness (see Fig. 4)
- $H_b$: Offset along $x_b$ axis between revolute joints on a link (see Fig. 4)
- $J$: Instantaneous Jacobian of the spatial linkage
- $K_{y^0}$: $n^{th}$ coefficient of the potential energy derivative equations for the linkage (includes link mass and geometry terms but no orientation dependant terms). The $y$ subscript denotes only components of potential energy in $y$ direction.
- $L_b$: Offset along the $x_b^*$ axis between revolute joints on a link (see Fig. 4)
- $U$: Linkage potential energy
- $cn$, $sn$: $\cos(n)$ and $\sin(n)$
- $x$, $y$, $z$: Cartesian end effector coordinates
- $x_b^*$, $z_b^*$: Coordinate axes associated with the first revolute joint encountered on a link when progressing from a lower numbered link to a higher numbered link in the chain (see Fig. 4).
- $A$, $B$, $\Gamma$: Three DC motor actuators in the haptic device
- $a$, $\beta$, $\gamma$: Axes associated with actuators $A$, $B$, and $\Gamma$, also represents angle of actuator rotation for each axis

REFERENCES


