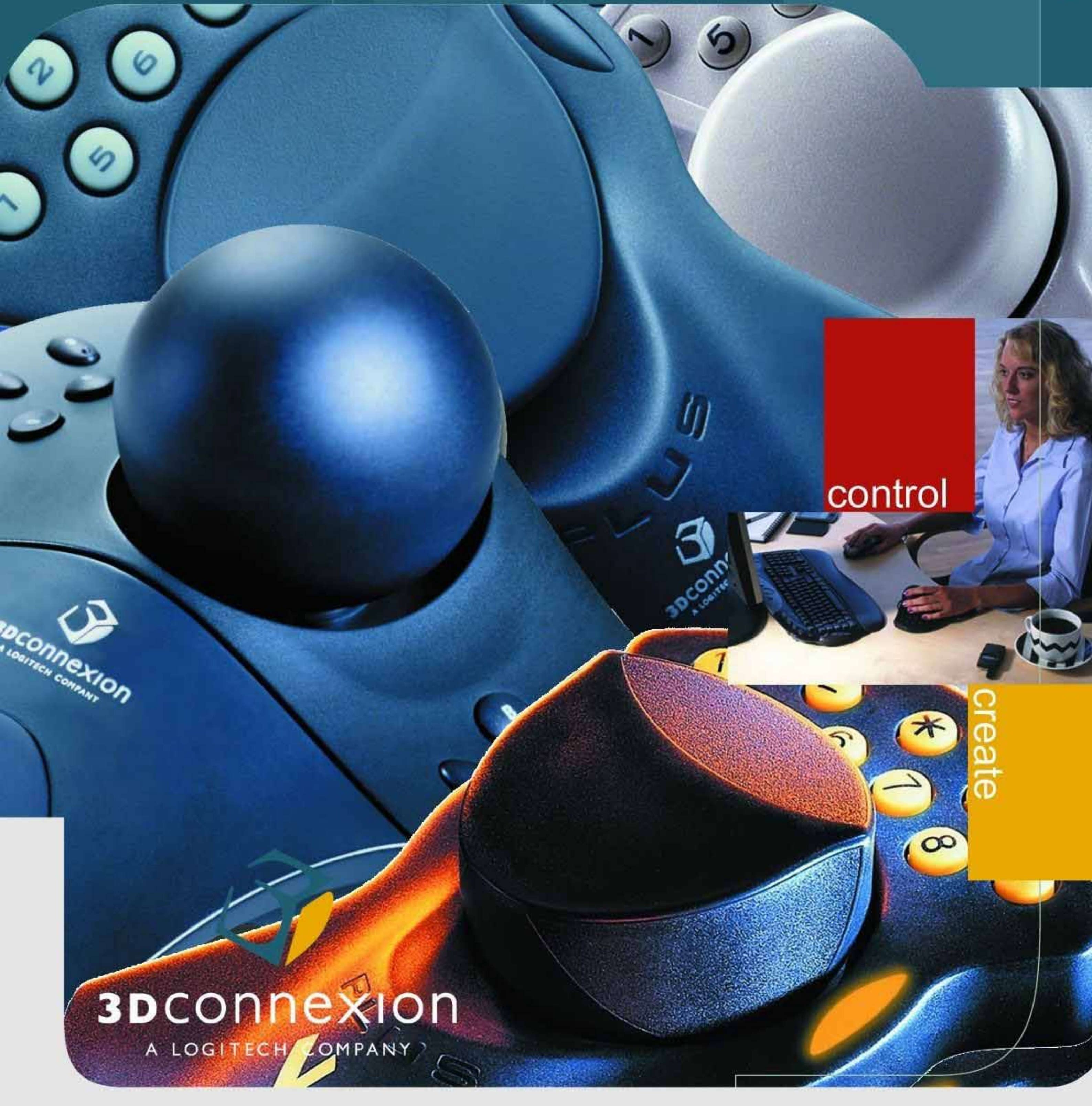
3D Motion Controller Serial Protocols

WWW.3DCONNEXION.COM



Copyright

Edited 04/2002 by 3Dconnexion - A Logitech Company.

NOTE: For optimal viewing of this document, it is recommended to use the latest version of Adobe Acrobat Reader, available on the 3DxWare driver CD-ROM or at www.adobe.com/acrobat.

Contents

Introduction 4

SpaceMouse Serial Protocol 5

Communication and Notation 5
Commands 5
Data Structures...9

Hints for SpaceMoue Serial Communication...................................11

Transmitting the First Command................11
Checking the Handshake Signals................11
Echo Mode OFF.......................................11
Carriage Return Character "\r"...................11
Fixed Number of Characters in Commands ..11

SpaceBall 4000 Serial Protocol............12

Introduction..12
Communication Parameters........................12
SpaceBall Packet Structure........................12
Packet Reference 12
Send Packets 12
Receive Packets 14
Configuration Packets 15
Other Packets 16
Escape Characters...................................16

Appendices 17

Turbo SpaceMouse...................................17
Mathematics of 3D Motion Control 18
Product Specifications...............................21

3Dconnexion Support..........................22

3Dconnexion Services 23

Warranty Information 24

3Dconnexion's Limited Lifetime Product Warranty 24
FCC Compliance Statement 24
European Economic Community Declaration of Conformance (CE)....................................25
VCCI Class B Declaration...........................25
Korean Class B Declaration........................25

Software License Agreement...............26

Introduction

This document details the process and protocols for communicating directly with the SpaceMouse and SpaceBall hardware. This process should only be used in the situations where the standard driver and API cannot meet the development needs. The Software Developers Kit for Unix and Windows is available on the 3Dconnexion website at: http://www.3dconnexion.com/software/sdk or on the CDROM in the */SDK* directory.

For basic information on how to install, use and configure your 3D Motion Controller, please review the 3D Motion Controller User's Manual. It is included on the 3Dconnexion CD-ROM and is also available on our website at http://www.3dconnexion.com/docs/.

Communication and Notation

The SpaceMouse exchanges data with the computer in the form of packets. Each packet starts with a character denoting the packet type, followed by a certain amount of useful data. As a general rule, four bits of useful data (which together make up a *nibble*) are coded into a byte. Coding is based on the scheme shown in the table below and always guarantees transmission of an even number of 1-bits in the byte. A parity-check of each byte is therefore possible and transmission errors of one bit can be detected.

Nibble Code	***4 Bits***	***Character***	***Character Hexadecimal***
0	0000	0	30H
1	0001	A	41H
2	0010	B	42H
3	0011	3	33H
4	0100	D	44H
5	0101	5	35H
6	0110	6	36H
7	0111	G	47H
8	1000	H	48H
9	1001	9	39H
A	1010	:	3AH
B	1011	K	4BH
C	1100	<	3CH
D	1101	M	4DH
E	1110	N	4EH
F	1111	?	3FH

To describe the SpaceMouse commands, this manual uses the notation *<nibbleX>*. It denotes a byte that contains four useful bits in the lower nibble (the second half of the byte), coded according to the table above. The individual bits of the byte are given in the first line by *<B7>* through *<B0>*, while the meaning of the bits is defined in the subsequent line. For example, suppose a bit sequence is defined as follows:

```
<B7><B6><B5><B4> <B3><B2><B1><B0>
                 <K4><K3><K2><K1>
```

Bits *<B7>* through *<B4>* must be set according to the nibble code (e.g. 0011 = 3, 0100 = 4), while bit *<B3>* contains the state of key *<K4>*, *<B2>* the state of key *<K3>*, etc. Each command must be terminated by a carriage return character, "\r".

Commands

Commanding the SpaceMouse is done by sending the device one of the commands described below.

Keyboard Command

Function:	Transmits the current state of the keys. Occurs any time a key is pressed or released.	
Command:	kQ\r	
Returns:	k<nibble1><nibble2><nibble3>\r	
<nibble1>	<B7...4>	<B3><B2><B1> <B0> <K4><K3><K2> <K1>
<nibble2>	<B7...4>	<B3><B2><B1> <B0> <K8><K7><K6> <K5>
<nibble3>	<B7...4>	<B3><B2><B1> <B0> <Q><-><+><K*>

For example, if key 6 is pressed, SpaceMouse transmits the packet "k0B0\r" to the computer. If the key is released, SpaceMouse transmits the packet "k000\r" indicating that no key is any longer being pressed.

Mode Set Command

Function:	Defines the operating mode and the structure of the data packets.	
Command:	m<nibble>\r	
<nibble>	<B7...4> < B3 > < B2 > <B1> <B0> <mouse> <dom> <tra> <rot>	
<mouse>	= 0	Should be set to 0 (obsolete).
<dom>	= 1	All components in the data packet are set to zero except the component with the largest magnitude.
		of the data packet.
<tra>	= 1	Has no effect on the components of the data packet.
	= 0	The translational components of the data packet (inputs X, Y and Z) are set to zero.
<rot>	= 1	Has no effect on the components of the data packet.
	= 0	The rotational components of the data packet (inputs A, B and C) are set to zero.
Returns:	The selected mode (in the same format as the command).	

For example, the packet "m6\r" instructs the SpaceMouse to operate in 3D mode and to transmit only the translational component with the largest magnitude.

Mode Status Command

Function:	Interrogates the current operating mode of the SpaceMouse.
Command:	mQ\r
Returns:	m<nibble>\r
<nibble>	<B7...4> < B3 > < B2 > <B1> <B0> <mouse> <dom> <tra> <rot>

See complete nibble description in previous section, Mode Set Command.

Compress Mode Set Command

Function:	Defines the extended operating mode of the SpaceMouse and the structure of the data packets.	
Command:	c<nibble1><nibble2>\r	
<nibble1>	<B7...4> < B3 > < B2 > <B1> <B0> <mouse> <dom> <tra> <rot>	
<mouse>	Should be set to 0 (obsolete).	
<dom>	= 1	All components in the data packet are set to zero except the component with the largest magnitude.
		of the data packet.
<tra>	= 1	Has no effect on the components of the data packet.
	= 0	The translational components of the data packet (inputs X, Y and Z) are set to zero.
<rot>	= 1	Has no effect on the components of the data packet.
	= 0	The rotational components of the data packet (inputs A, B and C) are set to zero.
<nibble2>	<B7...3> < B2 > < B1 > < B0 > <quicktip><extkey><compress>	
<quicktip>	= 1	Quicktip is enabled.
	= 0	Quicktip is disabled.
<extkey>	= 1	State of the SpaceMouse plus (+) button appears in bit *<B1>* of *<nibble3>* of the keyboard response (command "kQ\r"). State of the minus (-) button appears in bit *<B2>* of *<nibble3>*.
	= 0	State of the SpaceMouse plus (+) button appears in bit *<B1>* of *<nibble2>* of the keyboard response (command "kQ\r"). State of the minus (-) button appears in bit *<B2>* of *<nibble2>*.
<compress>	= 1	Data packets are transmitted in the Turbo SpaceMouse format. Note that the baud rate does not change; the SpaceMouse still sends data at 9600 baud. The maximum data packet transmission rate is 40 ms.
	= 0	Data packets are transmitted in the standard SpaceMouse format.
Returns:	The selected compressed mode (in the same format as the command).	

Compress Mode Status Command

Function:	Interrogates the current operating mode of the SpaceMouse.
Command:	cQ\r
Returns:	c<nibble1><nibble2>\r
<nibble1>	<B7...4> < B3 > < B2 > <B1> <B0> <mouse> <dom> <tra> <rot>
<nibble2>	<B7...3> < B2 > < B1 > < B0 > <quicktip><extkey><compress>

See complete nibble descriptions in previous section, Compress Mode Set Command.

Data Request Command

Function:	Requests data packets from the SpaceMouse.
Command:	dQ\r
Returns:	d <X3><X2><X1><X0> <Y3><Y2><Y1><Y0> <Z3><Z2><Z1><Z0> <A3><A2><A1><A0> <B3><B2><B1><B0> <C3><C2><C1><C0>\r

See complete description of data structure in the related section Standard Data Structure.

Data Rate Setup Command

Function:	Defines the maximum and minimum time periods.	
Command:	p<nibble1><nibble2>\r	
<nibble1>	<B7...4>	<B3><B2><B1><B0> < max period >
<nibble2>	<B7...4>	<B3><B2><B1><B0> < min period >
<max period>	= 0...15	The maximum time period is given in milliseconds by the formula: (<max period> + 1) * 20
<min period>	= 0...15	The minimum time period is given in milliseconds by the formula: (<min period> + 1) * 20
Returns:	The selected data rate (in the same format as the command).	

For example, the packet "p?B\r" sets the maximum time period to 320 ms and the minimum time period to 60 ms. Note that SpaceMouse is not able to transmit data packets with a time period shorter than 40 ms (older devices only 60 ms).

Data Rate ? Command

Function:	Interrogates the selected maximum and minimum time periods.	
Command:	pQ\r	
Returns:	p<nibble1><nibble2>\r	
<nibble1>	<B7...4>	<B3><B2><B1><B0> < max period >
<nibble2>	<B7...4>	<B3><B2><B1><B0> < min period >

See complete nibble descriptions in previous section, Data Rate Setup Command.

Zeroing Command

Function:	Defines a new zero at the current position of the cap. All subsequent translational and rotational values are relative to this position.
Command:	z\r
Returns:	z\r

Note that the SpaceMouse does not transmit any further data until the cap is moved again.

Sensitivity Setup Command

Function:	Sets the sensitivity of the SpaceMouse. Defines relationships between 1) translational displacements of the cap and the corresponding translational data sent to the computer, and 2) rotational displacements of the cap and the corresponding rotational data sent to the computer.	
Command:	q<nibble1><nibble2>\r	
<nibble1>	<B7...4>	<B3><B2><B1><B0> < sensitivity tra >
<nibble2>	<B7...4>	<B3><B2><B1><B0> < sensitivity rot >
<sensitivity tra>	= 0 = 1...15	The relationship is linear. A corresponding ballistic (quadratic) function is used.
<sensitivity rot>	= 0 = 1...15	The relationship is linear. A corresponding ballistic (quadratic) function is used.
Returns:	The selected sensitivity (in the same format as the command).	

For example, the packet "q00\r" defines the sensitivity of both the translation and rotation as purely linear. Note that the output values are in the approximate range of ±400. The ballistic functions may be estimated as follows:

$$\begin{pmatrix} ouput \\ value \end{pmatrix} = 2^* < sensitivity > ^* \begin{pmatrix} input \\ displacement \end{pmatrix}$$

This command has an impact only to the values that are transmitted in the SpaceMouse standard format.

Sensitivity Status Command

Function:	Interrogates the current sensitivity values of the SpaceMouse.	
Command:	qQ\r	
Returns:	q<nibble1><nibble2>\r	
<nibble1>	<B7...4>	<B3><B2><B1><B0> < sensitivity tra >
<nibble2>	<B7...4>	<B3><B2><B1><B0> < sensitivity rot >

See complete nibble descriptions in previous section, Sensitivity Setup Command.

Null Radius Setup Command

Function:	Defines the null radius of the SpaceMouse.	
Command:	n<nibble>\r	
<nibble>	<B7...4>	<B3><B2><B1><B0> < null radius >
<null radius>	= 0...15	The smallest movement of the cap necessary to generate a nonzero value is defined, where zero requires the smallest movement and 15 requires the largest.
Returns:	The selected null radius (in the same format as the command).	

For example, The packet "nH\r" sets the null radius to 8. At this setting, movements of the cap (both translational and rotational) from its center position that correspond to about 2% or less of the cap's maximal displacement range will only generate values of zero.

Null Radius Status Command

Function:	Interrogates the current null radius setup.	
Command:	nQ\r	
Returns:	n<nibble>\r	
<nibble>	<B7...4>	<B3><B2><B1><B0> < null radius >

See complete nibble description in previous section, Null Radius Setup Command.

Beep Command

Function:	Activates (or deactivates) the internal beeper of the SpaceMouse for the specified amount of time.	
Command:	b<nibble>\r	
<nibble>	<B7...4>	< B3 > <B2><B1><B0> <on/off> < duration >
<on/off>	= 1	SpaceMouse beeps for the specified duration.
		specified duration.
<duration>	= 7	2000 milliseconds
	= 6	1500 milliseconds
	= 5	1000 milliseconds
	= 4	500 milliseconds
	= 3	250 milliseconds
	= 2	125 milliseconds
	= 1	64 milliseconds
	= 0	32 milliseconds
Returns:	b\r	

For example, the command "b<\r" activates the beeper for half a second.

Flash Command

Function:	Activates the internal flasher of the SpaceMouse for the specified amount of time.	
Command:	f<nibble>\r	
<nibble>	<B7...4>	< B3 > <B2><B1><B0> < S/R > < duration >
<S/R>	= 1	The SpaceMouse flasher is set to a simple flash mode for the specified duration (all LEDs are illuminated simultaneously).
		runlight mode for the specified duration (the LEDs are illuminated in sequence).
<duration>	See description of *<duration>* field in previous section, Beep Command.	
Returns:	f\r	

Light Set Command

Function:	Controls the operation of the LEDs.	
Command:	l<nibble1><nibble2><nibble3>\r	
<nibble1>	<B7...3> <B2>< B1 >< B0 > <red><right yellow><left yellow>	
<red>	= 1	Turns the red LEDs on.
<right yellow>	= 1	Turns the right yellow LED on.
	= 0	Turns the right yellow LED off.
<left yellow>	= 1	Turns the left yellow LED on.
	= 0	Turns the left yellow LED off.
<nibble2>	Currently reserved for future development.	
<nibble3>	Currently reserved for future development.	
Returns:	The selected status of the LEDs (in the same format as the command).	

Light Status Command

Function:	Interrogates the current status of the LEDs.
Command:	lQ\r
Returns:	l<nibble1><nibble2><nibble3>\r
<nibble1>	<B7...3> <B2>< B1 >< B0 > <red><right yellow><left yellow>

See complete nibble descriptions in previous section, Light Set Command.

Version Command

Function:	Interrogates the version of the installed firmware.
Command:	vQ\r
Returns:	A string containing the version of the installed firmware.

The following is an example string returned:

v MAGELLAN Version 5.49 by LOGITECH INC. 10/22/96

Data Structures

Data packets are transmitted from the SpaceMouse to the computer using the data structures described below. Note that these packet structures cannot be used as commands.

Standard Data Structure

Structure:	d <X3><X2><X1><X0> <Y3><Y2><Y1><Y0> <Z3><Z2><Z1><Z0> <A3><A2><A1><A0> <B3><B2><B1><B0> <C3><C2><C1><C0>\r
<X,Y,Z,A,B,C 3,2,1,0>	The data packet transmits each of the six 16-bit inputs (X, Y, Z, A, B and C) coded into four nibbles. The higher-order nibble is transmitted first, followed by the lower-order nibble. The following formula is used to calculate each of the six inputs: <input> = <input3> * 4096 + <input2> * 256 + <input1> * 16 + <input0> * 1 – 32768

As a rule, three translational values and three rotational values are transmitted in the data packet INDEPENDENT of any mode settings (which utilize the mode command "m..\r"). The data packets are transmitted automatically if and only if the following conditions are met:

1) The data packet contains nonzero values. If all translational and rotational values are zero, only one data packet is transmitted. Further data packets are not transmitted until nonzero values appear in the data packet.

2) The maximum programmed time period is exceeded. The maximum time period is set with the data rate command "p..\r". If this period is exceeded, the SpaceMouse transmits a data packet without request.

3) The minimum time period is exceeded and a data packet has been requested via the data command "dQ\r". If data packets are requested more often than the minimum time period permits, the requests are ignored.

As an example, suppose the SpaceMouse transmits the following data packet:

dHBA5G?HKH000H0A6GNA6H06B\r

The corresponding values are as follows.

Input	*Characters*	*Decimal*	***Calculated Input Value***
X	HBA5	8,2,1,5	**533**
Y	G?HK	7,15,8,11	**-117**
Z	H000	8,0,0,0	**0**
A	H0A6	8,0,1,6	**22**
B	GNA6	7,14,1,6	**-490**
C	H06B	8,0,6,2	**98**

Error Message Structure

Structure:	e<nibble1><nibble2><nibble3>\	
<nibble1>	= 1	SpaceMouse has received an illegal command byte, which is returned to the computer with the parameters *<nibble2>* and *<nibble3>*. After transmission of the error message the SpaceMouse is ready to receive a new command.
	= 2	SpaceMouse has detected a framing error after receiving a character. If the character may be transmitted with only one stop bit, the parameters *<nibble2>* and *<nibble3>* have no meaning.

For example, if the SpaceMouse receives an illegal "C" command byte, it transmits the message "eAD3\r" to the computer. The two parameters "D" and "3" correspond to the nibbles 43H (see the nibble coding table). In ASCII code the value 43H corresponds to the upper-case "C" character.

Hints for SpaceMouse Serial Communication

The following hints should be helpful for developing drivers using SpaceMouse.

Transmitting the First Command

After receiving the first valid 3D command via the serial port (9600 Baud, 8 data bits, 2 stop bits), the SpaceMouse is in the 3D mode with a data transmission rate of 9600 Baud. All errors received on the serial port should be ignored until a valid byte is transmitted to the SpaceMouse.

Checking the Handshake Signals

Never transmit more than five bytes to SpaceMouse without checking the handshake signal (CTS) status. Loss of data and maladjustments of the SpaceMouse may occur if more than five bytes are transmitted without checking the status.

Echo Mode OFF

Some computers retransmit received characters (echo mode). This feature must be turned off when using the SpaceMouse. If echo mode is not turned off, more than five characters might be transmitted to the SpaceMouse without checking the handshake signal status. In addition, SpaceMouse will erroneously interpret each echo as a command.

Carriage Return Character "\r"

The commands transmitted to SpaceMouse must be terminated by the carriage return character "\r" ("\r" = CR = 13d = 0DH). If this character is missing at the end of a command string, the SpaceMouse will remain in a completely passive state while waiting for the terminating "\r". During this time no displacements of the cap or keyboard commands will be registered. If the SpaceMouse is fully passive and does not react to displacements of the cap or keyboard commands, it indicates the transmission of an erroneous command without a terminating "\r" character. (Keyboard commands always lead to reactions of the SpaceMouse unless a command without a terminating "\r" character is received. However, reactions to displacements of the SpaceMouse cap are transmitted only if the translation and rotation are in the ON state.)

Fixed Number of Characters in Commands

All SpaceMouse commands use a fixed number of characters. Transmitting a command with fewer characters to the SpaceMouse causes the device to wait for the terminating "\r" character. After each correctly received and interpreted command, the SpaceMouse returns a well-defined response to the computer, allowing the computer to check whether the command was correctly interpreted.

SpaceBall 4000 Serial Protocol

Introduction

This section is the reference for the SpaceBall 4000 packet protocol. It defines the structure and format of information transferred between the host computer and the SpaceBall device.

Communication Parameters

The SpaceBall 4000 communicates using an RS-232C interface. The communication parameters used by the device are:

- 9600 baud
- 8 data bits
- no parity
- one stop bit

The SpaceBall 4000 uses software (Xon/Xoff) and hardware (CTS/RTS) flow control. In order to avoid a serial deadlock, the SpaceBall will automatically resume transmission 1.5 seconds after an Xoff is received. During normal operation, the SpaceBall will only Xoff the host during a reset.

Only five of the RS-232 lines are used by the SpaceBall: transmit, receive, ground, ready to send (RTS) and data terminal ready (DTR).

There are two kinds of SpaceBall packets: data and request. Both packets utilize the same packet structure. The device responds to all host requests.

SpaceBall Packet Structure

A SpaceBall packet structure is composed of three elements: a header byte, zero or more data bytes and a packet terminator.

- The packet header is a printable ASCII character.
- Data bytes vary by the type of header.
- For send packets, the packet terminator is an ASCII <u>return</u> character (hexadecimal value of 0x0D).
- For receive packets, the packet terminator is an ASCII <u>return</u> character (hexadecimal value of 0x0D).

Basic Packet Structure Examples

Packet type	Byte values (hex)	ASCII equivalent
Send	53 0D	S<RETURN>
Receive	53 0D	S<RETURN>

During command parsing, the SpaceBall 4000 firmware reads the header byte and subsequent data bytes (depending on the header byte encountered), any extra bytes received prior to a packet terminator are ignored (except for the special serial control characters Xon and Xoff). Any packet prematurely terminated by a carriage return is discarded. The device responds to every host command when received.

Packet Reference

This section lists the various data packet types and their format. Within the listing, each character represents a single byte of data. Bold values represent variable-data fields. For example "*****aabbccddeeff**" would refer to a 13 byte data packet with '*' as the header byte and 12 bytes of data.

Send Packets

These are packets sent from the computer to the SpaceBall.

Emit Patterned Beep

Description:	Causes the firmware to emit a pattern of beeps.
Packet Header:	B (0x42)
Packet Data:	Up to 16 upper and lower case letters.
Example Packet:	BcCcC\r (Sends a double beep.)
Response:	None

The beep pattern is determined by the data packets within the beep string. Letters that are sooner in the alphabet stand for shorter beeps or pauses. Lower case letters mean beep on. Upper case letters mean beep off. The last character of this data packet should be a beep off.

Emit Single Beep

Description:	Generate a single beep.
Packet Header:	S (0x53)
Packet Data:	None.
Example Packet:	S\r
Response:	S\r

Enable Ball Data

Description:	Enables data packets.
Packet Header:	M (0x4D)
Packet Data:	None.
Example Packet:	M\r
Response:	M\r

The SpaceBall will not send any ball movement packets until this command is sent.

Note: After a power-up, the SpaceBall sends a "null-data" packet (data packet with all data equals zero) after the M\r.

Disable Ball Data

Description:	Disables data packets.
Packet Header:	- (0x2D)
Packet Data:	None.
Example Packet:	-\r
Response:	-\r

The SpaceBall will not send any ball movement packets after receiving this command. This is the default state.

Get Device Descriptor

Description:	This command requests the SpaceBall 4000-specific reset string.
Packet Header:	" (0x22)
Packet Data:	None.
Example Packet:	"\r
Response:	"1 Spaceball 4000 FLX\r "2 B:**BB H** PnP:**p** Az:**a** Sns:**s OOOO WW**\r "3 V**X.xx** created on **mmm-dd-yyyy**\r "4 Copyright(C) **YYYY** Spacetec IMC Corporation\r
BB	Number of buttons
H	Orientation (L for left, R for right)
p	0 if PnP is disabled, 1 if enabled.
a	0 if AutoZero is disabled, 1 if enabled.
s	Sensitivity type: S for standard, C for cubic
OOOO	AutoZero period in Hex (milliseconds)
WW	AutoZero window in raw reading units
X	Major firmware version
xx	Minor firmware version
mmm	Month firmware was created: (*Jan-Feb-Mar-Apr-May-Jun-Jul-Aug-Sep-Nov-Dec*)
dd	Day (0-31) firmware was created
yyyy	Year firmware was created

Get Rezero Ball

Description:	Rezero the SpaceBall.
Packet Header:	Z (0x5a)
Packet Data:	None.
Example Packet:	Z\r
Response:	Z 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00\r

Rezero takes the current position as the Powersensor rest position. All subsequent measurements (until reset) are taken relative to this position.

Get Device Information

Description:	Request the firmware version and build date.
Packet Header:	h (0x68)
Packet Data:	None.
Example Packet:	h\r
Response:	HvFirmware version 2.42 created on 24-Oct-1997\r

Cubic Sensitivity Enable/Disable

Description:	Select cubic or linear sensitivity curve.
Packet Header:	Y (0x59)
Packet Data:	C – To enable cubic sensitivity. S – To enable standard (linear) sensitivity.
Example Packet:	YC\r
Response:	YC\r

Enable/Disable Auto Rezero

Description:	Set the Auto Rezero state.
Packet Header:	A (0x41)
Packet Data:	E – Enable auto rezero D – Disable auto rezero
Example Packet:	AD\r
Response:	a**oooowwX**\r
oooo	AutoZero period (in hexadecimal).
ww	AutoZero window (in hexadecimal).
X	D (0x44) for disabled, E(0x45) for enabled.

Reset Device

Description:	This command causes the SpaceBall to RESET.
Packet Header:	@ (0x40)
Packet Data:	None.
Example Packet:	*@|r*
Response:	@1 Spaceball is alive and well after a software reset.\r @2 Firmware version 2.42 created on 24-Oct-1997.\r

Receive Packets

These are packets sent from the SpaceBall to the computer.

Device Reset

Description:	Power on greeting string.
Packet:	@1 Spaceball alive and well after a **AAAAAAA** reset.\r @2 Firmware version 2.42 created on 24-Oct-1997.\r
A…A	Type of reset: "poweron" "watchdog" "hardware" "software"

When power is applied to the device, it will perform its initialization. During this initialization the current position will be used as the Zero Position. The firmware will then wait a total of 2 seconds before sending the above string and emitting a double beep.

Ball Data

Description:	This packet contains movement data for the SpaceBall.
Packet Header:	D (0x44)
Packet Data:	D**ppxxyyzzrruuvv** (each letter stands for 1 byte)
pp	Period of data. This is the number of 1/16th milliseconds since the last ball data packet.
xx…zz	Translational forces.
rr…vv	Rotational forces (torque).

Whenever the ball is moved, a packet is generated by the SpaceBall. In order to receive a data packet, the "Enable Data" packet has to be sent first.

Actual force and torque values are 16-bit signed integers. The first byte is the higher significant byte of the 16 bit signed integer, the second byte is the lower significant byte. Decoding to a 16bit signed integer can be done like this:
signed int ForceX = (packet[3] << 8) | (packet[4]);

The resolution is 160 mNm and 4.4mNm for force and torque. This results in a force and torque range of +/- 20.48 N and +/- 0.5632 mNm. Positive x is to the right, +y is up and +z is upwards. Positive torque values are clockwise as viewed from the center of the ball out along the positive force axes.

Button Data

Description:	The button data packet is a bit field indicating the current status of the buttons and occurs whenever any button is pressed or released.
Packet Header:	K (0x4b)
Packet Data:	**aa**\r
aa	Backwards-compatible button status bits.

Button data follows the bit pattern below. The status of only eight of the buttons is transferred in these data bytes.

Upper Bits

7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	1	0	B8	0	B7	B6	B5

Lower Bits

7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	1	0	0	B4	B3	B2	B1

Advanced Button Data

Description:	The advanced button data packet is a bit field indicating the current status of the buttons and occurs whenever any button is pressed or released.
Packet Header:	. (0x2e)
Packet Data:	**bb**\r
bb	Expanded button status. It follows the bit pattern below.

This button status includes all twelve buttons and the lefty-bit, indicating the orientation of the device. If clear, the lefty-bit indicates that the device is in the left-handed mode (default).

Upper Bits

7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	1	Left	B12	B11	B10	B9	B8

Lower Bits

7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
B7	1	B6	B5	B4	B3	B2	B1

Error Data Packet

Description:	If one or more errors occur the SpaceBall generates this error packet.
Packet Header:	E (0x45)
aa	Bit code referring to the errors received.

Errors follow the bit pattern below. Notice, the reserved error bits for expansion.

Upper Bits

7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	1	0	0	0	E10	E9	E8

Lower Bits

7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	1	E6	E5	E4	E3	E2	E1

E1	Eclipse Register
E2	Eeprom checksum incorrect
E3	Eclipse timed out
E4	Transmit timeout
E5	Receive queue overflow
E6	Receive error
E8	Packet too long
E9	Packet Ignored
E10	Command unrecognized

Configuration Packets

The following packets affect the operation of the SpaceBall across sessions. These configuration parameters remain set even if power is cycled to the device, but won't take effect until the next device reset. In general, these packets will not be used but are provided as a backdoor to some of the inner workings of the device.

Change Automatic Zero parameters

Description:	Sets a flag in the EEPROM to enable automatic rezero.
Packet:	A (0x5b)
Packet Data:	ooooww\r
Packet Example:	A250010
Response:	A**ooooww**\r
oooo	AutoZero period (in hexadecimal). Set to zero to disable automatic zeroing capabilities.
ww	AutoZero window (in hexadecimal)

Note: Do not use the actual values for these parameters, but the ASCII encoded values, e.g., for a period of 0x2710 (10 seconds) and a window of 0x32, the packet should read "A271020".

Other Packets

There are other packets recognized by the SpaceBall 4000 that are not used in general practice and are included for compatibility with previous devices. The packets listed here have not changed from the x003C protocol.

Ignore Packet

Description:	The SpaceBall will ignore this entire packet.
Packet Header:	<SPACE> (0x20) or <TAB> (0x09)
Packet Data:	Anything.
Example Packet:	<SPACE>test\r
Response:	None

Echo Packet

Description:	Echo this packet. The device responds by returning the same packet received.
Packet Header:	% (0x25)
Packet Data:	Anything.
Example Packet:	%test\r
Response:	%test\r

Escape Characters

Some data packets will contain an additional "special character" ('^' – 0x5e) inserted within the data stream. This escape character notifies the system to modify the following byte. If the following byte is 'Q', (0x51), 'S' (0x53), or an 'M' (0x4d) the seventh bit should be cleared. (Which should result in 0x11, 0x13, 0x0d respectively.) If the following character is a second '^' (0x5e) then no modification should be made. (Do not forget to remove the '^' character from the packet buffer.)

Turbo SpaceMouse

The differences between the standard SpaceMouse (SSM) and the Turbo SpaceMouse (TSM) are the baud rates and the data packet formats. The TSM uses a double-speed internal clock and a shorter data packet format. The baud rate at the serial port is also doubled to 19200 baud (8 bit, no parity and two stop-bits). This allows the TSM a faster data packet transmission rate of 18 ms (as compared with 58 ms of the SSM). Note that the TSM uses the same command and message formats as the SSM, which have already been described in this document.

Communication

Six bits of useful data are coded into a byte. The equivalent 8-bit value (the first two bits set to zero plus the six bits of useful data) is coded in two nibbles, as shown in the table at right.

Data Structure

Data packets are transmitted from the TSM to the computer using the structure described below. Note that this packet structure cannot be used as a command.

Structure:	d <tx1><tx0> <ty1><ty0> <tz1><ty0> <ra1><ra0> <rb1><rb0> <rc1><rc0> <cs1><cs0>\r
<t,r x,y,z,a,b,c 1,0>	The data packet contains two bytes for each of the six input values (x, y, z, a, b and c). Only the lower six bits of each byte are used. Each pair of 6-bit bytes are combined into a 12-bit value. The high-order six bits are transmitted first and the low-order six bits are transmitted second. In general the following formula is used to calculate each of the six inputs: <t,r input> = <t,r input 1> * 64 + <t,r input 0> - 2048

6 Bits	*8 Bits*	*2-Nibble Code (Hex)*	*Character*
0	128	80	Ç
1	65	41	A
2	66	42	B
3	131	83	â
4	68	44	D
5	133	85	à
6	134	86	å
7	71	47	G
8	72	48	H
9	137	89	ë
10	138	8A	è
11	75	4B	K
12	140	8C	î
13	77	4D	M
14	78	4E	N
15	143	8F	Å
16	80	50	P
17	145	91	æ
18	146	92	Æ
19	83	53	S
20	148	94	ö
21	85	55	U
22	86	56	V
23	151	97	ù
24	152	98	ÿ
25	89	59	Y
26	90	5A	Z
27	155	9B	¢
28	220	DC	■
29	157	9D	¥
30	158	9E	₧
31	95	5F	_
32	224	E0	α
33	161	A1	í
34	162	A2	ó
35	99	63	c
36	164	A4	ñ
37	101	65	e
38	102	66	f
39	167	A7	°
40	168	A8	¿
41	105	69	i
42	106	6A	j
43	171	AB	½
44	108	6C	l
45	173	AD	¡
46	174	AE	«
47	111	6F	o
48	176	B0	░
49	113	71	q
50	114	72	r
51	179	B3	│
52	116	74	t
53	181	B5	╡
54	182	B6	╢
55	119	77	w
56	120	78	x
57	185	B9	╣
58	186	BA	║
59	123	7B	{
60	188	BC	╝
61	125	7D	}
62	62	3E	>
63	191	BF	┐

<cs 1,0>

The checksum is also transmitted in two bytes, with six significant bits per byte. The checksum is calculated with the following formula:

```
<cs> = <cs1> * 64 + <cs0>
```

For an error-free transmission, the checksum is equal to the sum of the transmitted data bytes of all six inputs. The TSM performs a check by calculating this value (each data byte is interpreted as an unsigned integer):

```
<cs> =  <tx1> + <tx0> +
        <ty1> + <ty0> +
        <tz1> + <tz0> +
        <ra1> + <ra0> +
        <rb1> + <rb0> +
        <rc1> + <tc0>
```

As an example, suppose the TSM transmits the following data packet:

```
dí╢¢qαÇ₧j₧║Zà▄B\r
```

The corresponding values are as follows.

Inpu		*Nibbles (Hex.)*	*Lower 6 Bits (Hex.)*	*Dec.*	***Calc'd Input Value***
x	í╢	A1,B6	21,36	33,54	**118**
y	¢q	9B,71	1B,3B	27,49	**-271**
z	αÇ	E0,80	20,00	32,0	**0**
a	₧j	9E,6A	1E,2A	30,42	**-86**
b	₧║	9E,BA	1E,3A	30,58	**-70**
c	Zà	5A,85	1A,05	26,5	**-379**
cs	▄B	DC,42	1C,02	28,2	**1794**

cs = A1 + B6 + 9B + 71 + E0 + 80 +
9E + 6A + 9E + BA + 5A + 85
= 702 *(Hex.)*
= **1794** *(Dec.)*

Mathematics of 3D Motion Control

This appendix outlines the mathematics necessary for describing arbitrary translational and rotational motion of an object. A cube serves as a good example for demonstrating the steps involved in the computations. Suppose the center of the cube is originally aligned with the origin of the xyz-coordinate system and its faces are parallel to the xy-, yz- and xz-planes. If the cube has an edge length of 2 units, its corners are given by the following set of eight points:

$P_{1\ old}$ (1, 1, 1)	$P_{2\ old}$ (-1, 1, 1)
$P_{3\ old}$ (-1, -1, 1)	$P_{4\ old}$ (1, -1, 1)
$P_{5\ old}$ (1, 1, -1)	$P_{6\ old}$ (-1, 1, -1)
$P_{7\ old}$ (-1, -1, -1)	$P_{8\ old}$ (1, -1, -1)

Note that a physical unit of length is not required.

One-Step Motion

If the cube is moved due to a translational or rotational displacement of the 3D Motion Controller handle, eight new points must be generated using the eight old points. To accomplish this, the device sends the six values X, Y, Z, A, B and C.

For the cube's translational motion, the values X, Y and Z have to be added to the original coordinates of the corner points. Thus a new point P_{new} is generated from an old point P_{old} using the equation

$$P_{new} = P_{old} + T_{XYZ}$$

Shown explicitly, this formula consists of the following three equations:

$$P_{new\ X} = P_{old\ X} + X$$
$$P_{new\ Y} = P_{old\ Y} + Y$$
$$P_{new\ Z} = P_{old\ Z} + Z$$

For the cube's rotational motion, the values A, B and C have to be incorporated into a 3x3 rotation matrix R.

$$R = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix}$$

The matrix elements are computed as follows:

$$R_{11} = (\cos A)(\cos B)$$
$$R_{12} = (\sin A)(\cos C) - (\cos A)(\sin B)(\sin C)$$
$$R_{13} = (\sin A)(\sin C) + (\cos A)(\sin B)(\cos C)$$
$$R_{21} = -(\sin A)(\cos B)$$
$$R_{22} = (\cos A)(\cos C) + (\sin A)(\sin B)(\sin C)$$
$$R_{23} = (\cos A)(\sin C) - (\sin A)(\sin B)(\cos C)$$
$$R_{31} = -(\sin B)$$
$$R_{32} = -(\cos B)(\sin C)$$
$$R_{33} = (\cos B)(\cos C)$$

Using this rotation matrix, the effects of rotation are calculated with the formula

$$P_{new} = [R](P_{old})$$

Shown explicitly, this formula consists of the following three equations:

$$P_{new\ X} = (R_{11})(P_{old\ X}) + (R_{12})(P_{old\ Y}) + (R_{13})(P_{old\ Z})$$
$$P_{new\ Y} = (R_{21})(P_{old\ X}) + (R_{22})(P_{old\ Y}) + (R_{23})(P_{old\ Z})$$
$$P_{new\ Z} = (R_{31})(P_{old\ X}) + (R_{32})(P_{old\ Y}) + (R_{33})(P_{old\ Z})$$

Combining the equations for translational and rotational motion yields the equation

$P_{new} = [R](P_{old}) + T_{XYZ}$

This equation describes the one-step motion of the graphical cube as represented by its eight corner points. Shown explicitly, this formula consists of the following three equations:

$P_{new\,X} = (R_{11})(P_{old\,X}) + (R_{12})(P_{old\,Y}) + (R_{13})(P_{old\,Z}) + X$

$P_{new\,Y} = (R_{21})(P_{old\,X}) + (R_{22})(P_{old\,Y}) + (R_{23})(P_{old\,Z}) + Y$

$P_{new\,Z} = (R_{31})(P_{old\,X}) + (R_{32})(P_{old\,Y}) + (R_{33})(P_{old\,Z}) + Z$

These equations do not change the size or shape of the cube (its edges are the same length as before and its corners are still right angles). In other words, after applying the above equations, we have a cube identical to the original but with a different position and orientation in space.

For example, suppose the controller delivers the following set of values:

X = 0.5	Y = 0	Z = -4.0
A = 0	B = 0	C = 0.3

The new position and orientation taken by the cube are computed by plugging the rotation matrix values and the original coordinates of the cube's eight corners into the combined equations for translation and rotation. The calculation is shown below for the first point, P_1.

$$R = \begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 0.955 & 0.296 \\ 0.0 & -0.296 & 0.955 \end{bmatrix}$$

$P_{1\,new\_X} = (1.0)(1) + (0.0)(1) + (0.0)(1) + 0.5 = 1.5$

$P_{1\,new\_Y} = (0.0)(1) + (0.955)(1) + (0.296)(1) + 0 = 1.251$

$P_{1\,new\_Z} = (0.0)(1) + (-0.296)(1) + (0.955)(1) - 4.0 = -3.340$

Similar calculations for the other seven points yield the following new set of points:

$P_{1\,new}$ (1.5, 1.251, -3.340)
$P_{2\,new}$ (-0.5, 1.251, -3.340)
$P_{3\,new}$ (-0.5, -0.660, -2.749)
$P_{4\,new}$ (1.5, -0.660, -2.749)
$P_{5\,new}$ (1.5, 0.660, -5.251)
$P_{6\,new}$ (-0.5, 0.660, -5.251)
$P_{7\,new}$ (-0.5, -1.251, -4.660)
$P_{8\,new}$ (1.5, -1.251, -4.660)

Note that the SpaceMouse/SpaceBall delivers [signed] integers that may be transformed into signed, floating-point decimals using unique scalings for translation and rotation. Thus the above set of example values is possible. The choice of an appropriate scaling factor for translation and another for rotation is dependent on the tasks to be performed and the available graphics and computational power of the computer.

Continual Motion

After this one-step motion of the cube, the graphics system continues to accept new values from the device, which are indicating that the cube's translational and rotational motions should continue. These new motions must be integrated into the above equations.

For translation this simply means adding the motions. The total translation after proceeding from step *n-1* (the previous step) to step *n* (the current step) is given by the following equations:

$$\sum_{i=1}^{n} X_i = \sum_{i=1}^{n-1} X_i + X_n$$

$$\sum_{i=1}^{n} Y_i = \sum_{i=1}^{n-1} Y_i + Y_n$$

$$\sum_{i=1}^{n} Z_i = \sum_{i=1}^{n-1} Z_i + Z_n$$

Note that $\sum_{i=1}^{n-1} X_i$, $\sum_{i=1}^{n-1} Y_i$ and $\sum_{i=1}^{n-1} Z_i$ are the translations summed up to step *n-1* and $\sum_{i=1}^{n} X_i$, $\sum_{i=1}^{n} Y_i$ and $\sum_{i=1}^{n} Z_i$ are the values sent to the computer by the 3D Controller in the current step, step *n*.

To calculate the total rotation, the successive rotation matrices must be multiplied. The values A, B and C are used to calculate the rotation matrix R_n for step *n* (just as described above for one-step motion). The rotation matrix R^*_{n-1}, which combines all previous rotations, is generated by successive multiplications of the rotation matrices up to the previous step, step *n-1*. (Matrices that are the result of successively multiplying all previous matrices will be denoted with a star [*].) This matrix represents all previous rotational motions. To compute the total rotation matrix R^*_n, all previous rotations (matrix R^*_{n-1}) must be combined with the rotation of the current step (matrix R_n). This consists of multiplying the two 3x3 matrices.

$R^*_n = [R_n][R^*_{n-1}]$

This yields a new 3x3 matrix whose elements are calculated using the following set of nine equations:

$R^*_{n\,11} = (R_{11})(R^*_{n-1\,11}) + (R_{12})(R^*_{n-1\,21}) + (R_{13})(R^*_{n-1\,31})$

$R^*_{n\,12} = (R_{11})(R^*_{n-1\,12}) + (R_{12})(R^*_{n-1\,22}) + (R_{13})(R^*_{n-1\,32})$

$R^*_{n\,13} = (R_{11})(R^*_{n-1\,13}) + (R_{12})(R^*_{n-1\,23}) + (R_{13})(R^*_{n-1\,33})$

$R^*_{n\,21} = (R_{21})(R^*_{n-1\,11}) + (R_{22})(R^*_{n-1\,21}) + (R_{23})(R^*_{n-1\,31})$

$R^*_{n\,22} = (R_{21})(R^*_{n-1\,12}) + (R_{22})(R^*_{n-1\,22}) + (R_{23})(R^*_{n-1\,32})$

$R^*_{n\,23} = (R_{21})(R^*_{n-1\,13}) + (R_{22})(R^*_{n-1\,23}) + (R_{23})(R^*_{n-1\,33})$

$R^*_{n\,31} = (R_{31})(R^*_{n-1,11}) + (R_{32})(R^*_{n-1\,21}) + (R_{33})(R^*_{n-1\,31})$

$R^*_{n\,32} = (R_{31})(R^*_{n-1\,12}) + (R_{32})(R^*_{n-1\,22}) + (R_{33})(R^*_{n-1\,32})$

$R^*_{n\,33} = (R_{31})(R^*_{n-1\,13}) + (R_{32})(R^*_{n-1\,23}) + (R_{33})(R^*_{n-1\,33})$

The new rotation matrix R^*_n describes all successively-executed rotations up to the current step, step *n*.

Using the new equations for the accumulated translation and rotation, a single formula can be written that transforms the original points of the cube into their new positions.

$$P_{new} = [R^*_n](P_{old}) + \sum_{i=1}^{n} T_{XYZ\,i}$$

Shown explicitly, this formula consists of the following three equations:

$$P_{new\,X} = (R^*_{11})(P_{old\,X}) + (R^*_{12})(P_{old\,Y}) + (R^*_{13})(P_{old\,Z}) + \sum_{i=1}^{n} X_i$$

$$P_{new\,Y} = (R^*_{21})(P_{old\,X}) + (R^*_{22})(P_{old\,Y}) + (R^*_{23})(P_{old\,Z}) + \sum_{i=1}^{n} Y_i$$

$$P_{new\,Z} = (R^*_{31})(P_{old\,X}) + (R^*_{32})(P_{old\,Y}) + (R^*_{33})(P_{old\,Z}) + \sum_{i=1}^{n} Z_i$$

Note that $\sum_{i=1}^{n} X_i$, $\sum_{i=1}^{n} Y_i$ and $\sum_{i=1}^{n} Z_i$ are simply the sums of all the translation commands up to the current step *n*, and R^*_n is the total rotation matrix generated by multiplying all previous rotation matrices.

For example, suppose the controller has now sent the following second set of values to the computer:

X = 1.5	Y = 0	Z = 0
A = 0.2	B = 0	C = 0

Summing the translational motion gives the following values:

$$\sum_{i=1}^{n} X_i = 2.0 \qquad \sum_{i=1}^{n} Y_i = 0.0 \qquad \sum_{i=1}^{n} Z_i = -4.0$$

The rotation matrix for the current step, R_n, is found from the new values for A, B and C.

$$R_n = \begin{bmatrix} 0.980 & 0.199 & 0.0 \\ -0.199 & 0.980 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}$$

This matrix is multiplied by the combined rotation matrix for all previous steps, R^*_{n-1} (which corresponds to the values calculated for matrix R^*_n in the previous step of the cube example). This yields the new total rotation matrix R^*_n:

$$R^*_n = \begin{bmatrix} 0.980 & 0.190 & 0.059 \\ -0.199 & 0.936 & 0.290 \\ 0.0 & -0.296 & 0.955 \end{bmatrix}$$

The new coordinates of the cube's corners are now calculated using this total rotation matrix, the summed translational motions and the ORIGINAL coordinates (NOT the coordinates calculated in the previous step). As an example the calculations for P_1 are shown below.

$P_{1\,new\_X} = (0.980)(1) + (0.190)(1) + (0.059)(1) + 2.0 = 3.229$

$P_{1\,new\_Y} = (-0.199)(1) + (0.936)(1) + (0.290)(1) + 0 = 1.027$

$P_{1\,new\_Z} = (0.0)(1) + (-0.296)(1) + (0.955)(1) - 4.0 = -3.340$

After these two steps, the corners of the cube are located at the following coordinates:

$P_{1\,new}$ (3.229, 1.027, -3.340)
$P_{2\,new}$ (1.268, 1.425, -3.340)
$P_{3\,new}$ (0.889, -0.448, -2.749)
$P_{4\,new}$ (2.849, -0.845, -2.749)
$P_{5\,new}$ (3.111, 0.448, -5.251)
$P_{6\,new}$ (1.151, 0.845, -5.251)
$P_{7\,new}$ (0.771, -1.027, -4.660)
$P_{8\,new}$ (2.732, -1.425, -4.660)

If the 3D Controller continues to send values, these computational steps must be repeated recursively.

Product Specifications

Feature/Specification	*SpaceMouse Classic*	*SpaceMouse Plus & Plus XT*	*SpaceBall 4000FLX*
Number of freely programmable buttons	9	11	12
Software-controllable keyboard LEDs	No	Yes (2 yellow, 1 red) XT Only	No
Quicktip virtual button	Yes	Yes	No
Device weight (for stability)	0.660 kg	0.680 kg	0.650 kg
Human Interface Form-Factor	Round Puck	Ergonomic Puck	Soft Touch Ball
Operating humidity (non-condensing)	10 to 98% RH	10 to 98% RH	10 to 98% RH
Operating temperature	+5 to +60 °C	+5 to +60 °C	+10 to +40 °C
Storage humidity	10 to 98% RH	10 to 98% RH	10 to 98% RH
Storage temperature	-40 to +85 °C	-40 to +85 °C	+6 to +60 °C
Supported systems	*UNIX:* DEC, HP, IBM, SGI, SUN, Linux *PC:* Win98, ME, WinNT, 2000, XP	*UNIX:* DEC, HP, IBM, SGI, SUN, Linux *PC:* Win98, ME, WinNT, 2000, XP	*UNIX:* DEC, HP, IBM, SGI, SUN, Linux *PC:* Win98, ME, WinNT, 2000, XP
Power source	5V / 9mA	5V / 9mA	5V / 10mA
Connector	Serial or USB	Serial or USB	Serial or USB
Baud Rate	9600 Baud	9600 Baud	9600 Baud
Standard Data Rate	40 ms	40 ms	50 ms
Compact Size L x W x H (mm)	165 x 112 x 40	188 x 129 x 44	213 x 152 x 76
Converter-adapters available for the following serial port connections	IBM 25-p D-Sub m IBM 9-p D-Sub m SGI 8-p mini-DIN f SGI 8-p DIN f SGI 9-p D-Sub f SUN 25-p D-Sub f	IBM 25-p D-Sub m IBM 9-p D-Sub m SGI 8-p mini-DIN f SGI 8-p DIN f SGI 9-p D-Sub f SUN 25-p D-Sub f	IBM 25-p D-Sub m IBM 9-p D-Sub m SGI 8-p mini-DIN f SGI 8-p DIN f SGI 9-p D-Sub f SUN 25-p D-Sub f
FCC, TUV/GS, VCCI, CSA or UL, & CE - Approved	Yes	Yes	Yes
Length of manufacturer's warranty	3 years	3 years	3 years
Standard driver source freely available	Yes	Yes	Yes
Unix Specific Features			
Dialbox Simulation	Yes	Yes	Yes
LPFK Simulation	Yes	Yes	Yes
Dominant Mode	Yes	Yes	Yes

Product Specifications
Please visit our website to see additional information on our complete line of 3D Motion Controllers at:
http://www.3dconnexion.com/products

3Dconnexion Support

If you have any questions or comments about your 3D Motion Controller, please contact the appropriate regional support center listed for your area.

When you call technical support, please be at your computer so that we can assist you. Please have the following information available when you call:

- Your name, company name, and telephone number
- Product name and version number
- Your computer configuration: CPU type, speed, memory, pointing device, video card (its memory and resolution)
- The platform and operating system you are running
- Your application name and version
- The version of 3d motion controller driver you are using

You can also send an email or a detailed fax. Clearly state your problem and include the information listed above.

Various information about 3Dconnexion's 3D Motion Controllers, including the latest driver versions, can be found at:
http://www.3Dconnexion.com

US / Asia / Americas

3Dconnexion
30600 Telegraph Road, Suite 4290
Bingham Farms, MI 48025
U.S.A.
Tel: + 1-248-331-1999
Fax: + 1-248-331-1399
Email: supportUS@3dconnexion.com
Web: www.3Dconnexion.com

Online Support
www.3Dconnexion.com/support/

Technical Support Hotline
+1 (888) 247 9545
(English Only)

Europe / EMEA

3Dconnexion GmbH
An der Hartmuehle 8
D-82229 Seefeld
Germany
Tel: + 49 (0) 8152-9919-0
Fax: + 49 (0) 8152-9919-50
Email: supportEU@3dconnexion.com
Web: www.3Dconnexion.com

Online Support
www.3Dconnexion.com/support/

Technical Support Hotline
+49 (8152) 9919 44
(English & German Only)

3Dconnexion Services

Please visit our website www.3Dconnexion.com for any of the following services.

New Integration
To find out details about integrating 3D motion controllers into your application you can contact one of the offices listed on the website.

Software Development Kit (SDK)
Is a documented C-based library available online for add-in development. Please visit our website for the latest information on our SDK at: http://www.3Dconnexion.com/software/sdk

Implementation support
To get help in your implementation efforts you can find support information on the website.

Application Support
A list of supported applications can be found at:
http://www.3Dconnexion.com/software/drivers

If a particular application is not on the list, please contact the software company producing that application to find out more about 3D Motion Controller support.

3Dconnexion's Limited Lifetime Product Warranty

Limited Warranty
3Dconnexion warrants that any hardware product accompanying this documentation shall be free from significant defects in material and workmanship for a period of three (3) years from the date of purchase. 3Dconnexion's limited warranty is nontransferable and is limited to the original purchaser. This warranty gives you specific legal rights, and you may also have other rights which vary under local laws.

Remedies
3Dconnexion’s entire liability and your exclusive remedy for any breach of warranty shall be, at 3Dconnexion’s option, to: (a) repair or replace the hardware, provided that the hardware is returned to the point of purchase or such other place as 3Dconnexion may direct, with a copy of the sales receipt, or (b) refund the price paid. Any replacement hardware will be warranted for the remainder of the original warranty period or thirty (30) days, whichever is longer. These remedies are void if failure of the hardware has resulted from accident, abuse, or misapplication.

DISCLAIMER OF WARRANTY
THE WARRANTIES EXPRESSLY SET FORTH IN THIS AGREEMENT REPLACE ALL OTHER WARRANTIES. 3DCONNEXION AND ITS SUPPLIERS EXPRESSLY DISCLAIM ALL OTHER WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD-PARTY RIGHTS WITH RESPECT TO THE HARDWARE. NO 3DCONNEXION DEALER, AGENT, OR EMPLOYEE IS AUTHORIZED TO MAKE ANY MODIFICATION, EXTENSION, OR ADDITION TO THIS WARRANTY. Some jurisdictions do not allow limitations on how long an implied warranty lasts, so the above limitation may not apply to you.

LIMITATION OF LIABILITY
IN NO EVENT WILL 3DCONNEXION OR ITS SUPPLIERS BE LIABLE FOR ANY COSTS OF PROCUREMENT OF SUBSTITUTE PRODUCTS OR SERVICES, LOST PROFITS, LOSS OF INFORMATION OR DATA, OR ANY OTHER SPECIAL, INDIRECT, CONSEQUENTIAL, OR INCIDENTAL DAMAGES ARISING IN ANY WAY OUT OF THE SALE OF, USE OF, OR INABILITY TO USE ANY 3DCONNEXION PRODUCT OR SERVICE, EVEN IF 3DCONNEXION HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN NO CASE SHALL 3DCONNEXION'S AND ITS SUPPLIERS’ TOTAL LIABILITY EXCEED THE ACTUAL MONEY PAID FOR THE 3DCONNEXION PRODUCT OR SERVICE GIVING RISE TO THE LIABILITY. Some jurisdictions do not allow the exclusion or limitation of incidental or consequential damages, so the above limitation or exclusion may not apply to you. The above limitations will not apply in case of personal injury where and to the extent that applicable law requires such liability.

FCC Compliance Statement

This device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions:

1) This device may not cause harmful interference.
2) This device must accept any interference received, including interference that may cause undesired operation.

NOTE: This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and receiver.

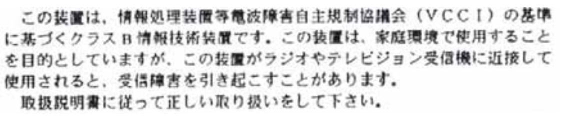
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- Consult the dealer or an experienced radio/TV technician for help.

CAUTION: The user is cautioned that changes or modifications to the equipment not expressly approved by the party responsible for compliance could void the user's authority to operate the equipment.

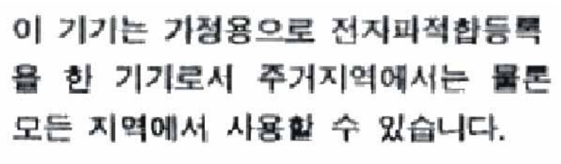
European Economic Community Declaration of Conformance (CE)

The SpaceMouse is attested to meet the essential protection requirements against electromagnetic emission, which are established in the regulations of the council for assimilating the rules and regulations of the member states about electromagnetic compatibility 89/336/EEC and changed by regulation 92/31 EEC. This declaration is valid for all samples produced according to the enclosed production drawings, which are part of this declaration. The following standards were used for judging the product concerning electromagnetic capability:
- For trouble emission: EN55022 edition: 05/95
- For trouble security: EN50082-1 edition: 03/93

VCCI Class B Declaration


この装置は、情報処理装置等電波障害自主規制協議会（ＶＣＣＩ）の基準に基づくクラスＢ情報技術装置です。この装置は、家庭環境で使用することを目的としていますが、この装置がラジオやテレビジョン受信機に近接して使用されると、受信障害を引き起こすことがあります。
取扱説明書に従って正しい取り扱いをして下さい。

Korea Class B Declaration


이 기기는 가정용으로 전자파적합등록을 한 기기로서 주거지역에서는 물론 모든 지역에서 사용할 수 있습니다.

Software License Agreement

3DCONNEXION

SOFTWARE LICENSE AGREEMENT

3DCONNEXION IS WILLING TO LICENSE THIS SOFTWARE TO YOU ONLY ON THE CONDITION THAT YOU ACCEPT ALL OF THE TERMS CONTAINED IN THIS LICENSE AGREEMENT. This is a legal agreement between you (either an individual end-user or an entity) and 3Dconnexion ("Agreement"). By using this software, you are agreeing to be bound by the terms and conditions of this Agreement. If you do not agree to the terms and conditions of this Agreement, promptly return the software and other items that are part of this product in their original package with your sales receipt to your point of purchase for a full refund, or if you have downloaded this software from a 3Dconnexion web site, then you must stop using the software and destroy any copies of the software in your possession or control.

1. Grant of Agreement. Subject to the terms and conditions of this Agreement, 3Dconnexion and its suppliers grant to you a nonexclusive license to use one copy of the software program and any documentation accompanying this Agreement ("Software") on one computer only with the 3Dconnexion product you have purchased. No other rights are granted. The Software is in use if it is loaded on the computer's permanent or temporary memory. For backup purposes only, you may make one copy of the Software. You must include on the backup copy all copyright and other notices included on the Software as supplied by 3Dconnexion. Installation on a network server for the sole purpose of your internal distribution of the Software is permitted only if you have purchased an individual Software license for each networked computer to which the Software is distributed.

2. Restrictions. The Software contains copyrighted material, trade secrets, and other proprietary materials of 3Dconnexion and its licensors. You agree that in order to protect those proprietary materials, except as expressly permitted by applicable law, neither you nor a third party acting on your behalf will: (i) decompile, disassemble or reverse engineer the Software; (ii) modify or create derivative works of the Software; (iii) use the Software in any manner to provide service bureau, commercial time-sharing or other computer services to third parties; (iv) transmit the Software or provide its functionality, in whole or in part, over the Internet or other network (except as expressly permitted above); or (v) sell, distribute, rent, lease, sublicense or otherwise transfer the Software to a third party, except upon a permanent transfer of the 3Dconnexion product using the Software; provided that: (a) all Software updates are included in the transfer, (b) you do not retain a copy of the Software, and (c) the transferee agrees to be bound by the terms and conditions in this Agreement.

3. Ownership. The Software is licensed, not sold, to you for use only under the terms and conditions of this Agreement, and 3Dconnexion reserves all rights not expressly granted to you in this Agreement. 3Dconnexion and/or its licensors retain title to the Software, and all intellectual property rights therein.

4. Termination. This Agreement is effective until terminated. Upon any violation of any of the provisions of this Agreement, rights to use the Software shall automatically terminate and the Software must be returned to 3Dconnexion or all copies of the Software destroyed. You may also terminate this Agreement at any time by destroying all copies of the Software in your possession or control. If 3Dconnexion makes a request via public announcement or press release to stop using the copies of the Software, you will comply immediately with this request. The provisions of paragraphs 3, 7, 8 and 12 will survive any termination of this Agreement.

5. Limited Product Warranty. 3Dconnexion warrants to you that the Software will substantially conform to its published documentation and the media containing the Software shall be free from defects in material, each for a period of ninety (90) days from the date of purchase. 3Dconnexion's limited warranty is nontransferable and is limited to the original purchaser. This warranty gives you specific legal rights, and you may also have other rights which vary under local laws.

6. Remedies. 3Dconnexion's entire liability and your exclusive remedy for any breach of warranty shall be, at 3Dconnexion's option, to: (a) repair or replace the Software or media, provided that the Software or media is returned to the point of purchase or such other place as 3Dconnexion may direct, with a copy of the sales receipt, or (b) refund the price paid. Any replacement Software or media will be warranted for the remainder of the original warranty period or thirty (30) days, whichever is longer. These remedies are void if failure of the Software or media has resulted from accident, abuse, or misapplication.

7. DISCLAIMER OF WARRANTY. THE WARRANTIES EXPRESSLY SET FORTH IN THIS AGREEMENT REPLACE ALL OTHER WARRANTIES. 3DCONNEXION AND ITS SUPPLIERS EXPRESSLY DISCLAIM ALL OTHER WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD-PARTY RIGHTS WITH RESPECT TO THE SOFTWARE OR MEDIA, AND ANY WARRANTIES OF NON-INTERFERENCE OR ACCURACY OF INFORMATIONAL CONTENT. NO 3DCONNEXION DEALER, AGENT, OR EMPLOYEE IS AUTHORIZED TO MAKE ANY MODIFICATION, EXTENSION, OR ADDITION TO THIS WARRANTY. SOME JURISDICTIONS DO NOT ALLOW LIMITATIONS ON HOW LONG AN IMPLIED WARRANTY LASTS, SO THE ABOVE LIMITATION MAY NOT APPLY TO YOU.

8. LIMITATION OF LIABILITY. IN NO EVENT WILL 3DCONNEXION OR ITS SUPPLIERS BE LIABLE FOR ANY COSTS OF PROCUREMENT OF SUBSTITUTE PRODUCTS OR SERVICES, LOST PROFITS, LOSS OF INFORMATION OR DATA, OR ANY OTHER SPECIAL, INDIRECT, CONSEQUENTIAL, OR INCIDENTAL DAMAGES ARISING IN ANY WAY OUT OF THE SALE OF, USE OF, OR INABILITY TO USE ANY 3DCONNEXION PRODUCT OR SERVICE, EVEN IF 3DCONNEXION HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN NO CASE SHALL 3DCONNEXION'S AND ITS SUPPLIERS' TOTAL LIABILITY EXCEED THE ACTUAL MONEY PAID FOR THE 3DCONNEXION PRODUCT OR SERVICE GIVING RISE TO THE LIABILITY. Some jurisdictions do not allow the exclusion or limitation of incidental or consequential damages, so the above limitation or exclusion may not apply to you. The above limitations will not apply in case of personal injury where and to the extent that applicable law requires such liability.

9. U.S. Government Restricted Rights. Use, duplication, or disclosure by the U.S. Government is subject to restrictions set forth in this Agreement and as provided in DFARS 227.7202-1(a) and 227.7202-3(a) (1995), DFARS 252.227-7013(c)(1)(ii) (OCT 1988) FAR 12.212(a) (1995), FAR 52.227-19, or FAR 52.227-14 (ALT III), as applicable. 3Dconnexion Inc. 180 Knowles Drive, Suite 100, Los Gatos, CA 95032.

10. Export Law Assurances. You agree and certify that neither the Software nor any other technical data received from 3Dconnexion will be exported outside the United States except as authorized and as permitted by the laws and regulations of the United States. If you have rightfully obtained the Software outside of the United States, you agree that you will not re-export the Software nor any other technical data received from 3Dconnexion, except as permitted by the laws and regulations of the United States and the laws and regulations of the jurisdiction in which you obtained the Software.

11. Agents and Third Party Purchasers. If you are acquiring the Software on behalf of another person or entity, you represent and warrant that you have the authority to bind the party or entity for which you are acquiring the Software to the terms and conditions of this Agreement.

12. General Terms and Conditions. This Agreement will be governed by and construed in accordance with the laws of the United States and the State of California, without regard to or application of its choice of law rules or principles. If for any reason a court of competent jurisdiction finds any provision of this Agreement, or portion thereof, to be unenforceable, that provision of the Agreement shall be enforced to the maximum extent permissible so as to affect the intent of the parties, and the remainder of this Agreement shall continue in full force and effect. This Agreement constitutes the entire agreement between the parties with respect to the use of the Software and supersedes all prior or contemporaneous understandings, communications or agreements, written or oral, regarding such subject matter. 3Dconnexion may, in its sole discretion, modify portions of this Agreement at any time. 3Dconnexion may notify you of any changes by posting notice of such modifications on 3Dconnexion's web site(s) or sending notice via e-mail, postal mail or other means. Your continued use of the Software following notice of such modifications shall be deemed to be your acceptance of any such modifications to the Agreement. If you do not agree to any such modifications, you must immediately stop using the Software and destroy all copies of the Software in your possession or control.

The Software is protected by United States copyright law and international treaty. Unauthorized reproduction or distribution of the Software is subject to civil and criminal penalties.