Magellan/SpaceMouse

Classic
Plus
Plus XT

Programmer's Guide

v.3.1

LogiCad3D
A LOGITECH COMPANY

Copyright

LogiCad3D GmbH - A Logitech Company
An der Hartmuehle 8
D-82229 Seefeld, Germany
Tel: + 49 (0) 8152-9919-0
Fax: + 49 (0) 8152-9919-50
Email: help@logicad3d.com
Web: www.logicad3d.com

The information in this manual is subject to change without notice. LogiCad3D shall not be held liable for technical or editorial errors or omissions contained herein, nor for incidental or consequential damages resulting from the furnishing, performance or use of this material. The information in this manual may not be changed without special notification. The instructions in this manual are checked regularly and necessary corrections are included in all subsequent editions. More copies or newer editions of this manual and technical information on the Magellan/SPACE MOUSE can be obtained only from LogiCad3D GmbH.

The citation of names in this manual as a rule does not mention existing patents, registered designs or trademarks. Missing corresponding remarks do not justify the assumption that the names may be freely usable. All trademarks are acknowledged where applicable.

NOTICE:
The use of the Magellan/SPACE MOUSE is primarily intended for graphical applications only. The company LogiCad3D is not liable for any damages (including all kinds of damage from lost profit, operating breakdown, loss of business information, data or other kinds of monetary loss) that are due to either proper or improper use of this LogiCad3D product. In any case, LogiCad3D's liability is restricted to the amount of money paid for the product. This exclusion does not hold for damages caused by LogiCad3D intentionally or grossly negligent. In the same way, claims based on general laws and rules of product liability remain untouched. For other applications LogiCad3D declines any liability or claims for damages.

SPACE MOUSE® is a registered European trademark of LogiCad3D GmbH.

Magellan™ is the US trademark of LogiCad3D Inc./GmbH.

This device uses one or more patents held by the Deutsches Zentrum für Luft- und Raumfahrt e.V. (DLR).

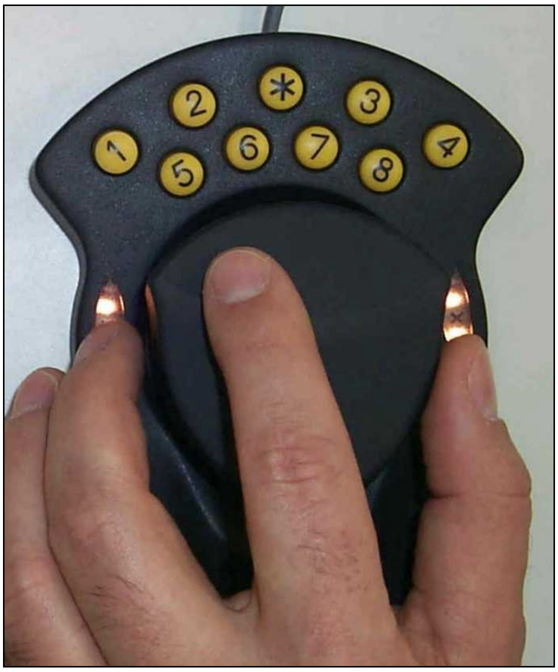
Edited 08/00 by LogiCad3D GmbH - A Logitech Company, GERMANY.

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

Contents

Magellan/SPACE MOUSE Operation

For basic information on how to install, use and configure the Magellan/SPACE MOUSE, please review the Magellan/SPACE MOUSE User's Manual. It is included on the LogiCad3D CD-ROM and is also available on our website at www.logicad3d.com/docs.



Manual Configuration

The various operating modes and sensitivities of the Magellan/SPACE MOUSE may be manually configured using the eight predefined key combinations described below. The keys must be pressed *simultaneously*. Press and hold the star key followed by the desired numerical key. For functions with on/off states, switching on is indicated by a double beep and switching off is indicated by a single beep.

K<*> and K<1>
Translation ON/OFF
Turns the translational degrees of freedom (inputs X, Y and Z) on or off. Turning translation off fixes the "screen position" of the onscreen object. Subsequently the Magellan/SPACE MOUSE sends an "m..\r" command without request. The default is ON.

K<*> and K<2>
Rotation ON/OFF
Turns the rotational degrees of freedom (inputs A, B and C) on or off. Turning rotation off fixes the orientation of the onscreen object. Subsequently the Magellan/SPACE MOUSE sends an "m..\r" command without request. The default is ON.

K<*> and K<3>
Dominant Mode ON/OFF
When dominant mode is on, only the input of the greatest magnitude is registered, i.e. the onscreen object moves in only one direction at a time. This can be a translational or rotational direction. Subsequently the Magellan/SPACE MOUSE sends an "m..\r" command without request. The default is OFF.

K<*> and K<4>
Zeroing
The Magellan/SPACE MOUSE is zeroed at the current position of the cap, as indicated by a double beep. All subsequent inputs are relative to this position. Subsequently the Magellan/SPACE MOUSE sends an "z\r" command without request.

K<*> and K<5>
Translation Sensitivity
The translational sensitivity is increased incrementally on a scale from zero (the default) to 7, as indicated by a single beep. Activating this key sequence the eighth time resets the translational sensitivity to zero, as indicated by a double beep. Subsequently the Magellan/SPACE MOUSE sends an "q..\r" command without request.

K<*> and K<6>
Rotation Sensitivity
The rotational sensitivity is increased incrementally on a scale from zero (the default) to 7, as indicated by a single beep. Activating this key sequence the eighth time resets the rotational sensitivity to zero, as indicated by a double beep. Subsequently the Magellan/SPACE MOUSE sends an "q..\r" command without request.

K<*> and K<7>
Zero Radius
The minimum displacement of the cap required to cause movement is increased incrementally on a scale from zero to 15 (the default is 13, as indicated by a single beep. After 15 the setting is reset to zero, as indicated by a double beep. Subsequently the Magellan/SPACE MOUSE sends an "n..\r" command without request.

K<> and K<8>*
Default Sensitivity

Returns the translational and rotational sensitivities and the zero radius to the default settings, as indicated by a double beep. Subsequently the Magellan/SPACE MOUSE sends an "nH\r" command and a "q00\r" command without request.

Communication and Notation

The Magellan/SPACE MOUSE 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 Magellan/SPACE MOUSE 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 Magellan/SPACE MOUSE 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> <K*>

For example, if key 6 is pressed, Magellan/ SPACE MOUSE transmits the packet "k0B0\r" to the computer. If the key is released, Magellan/SPACE MOUSE 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	Magellan/SPACE MOUSE is set into 3D mode.
<dom>	= 1	All components in the data packet are set to zero except the component with the largest magnitude.
	= 0	The bit has no effect on the components of the data packet.
<tra>	= 1	The bit 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	The bit 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 Magellan/SPACE MOUSE to operate in 3D mode and to transmit only the translational component with the largest magnitude.

Mode ? Command

Function:	Interrogates the current operating mode of the Magellan/ SPACE MOUSE.
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 Magellan/SPACE MOUSE and the structure of the data packets.	
Command:	c<nibble1><nibble2>\r	
<nibble1>	<B7...4> < B3 > < B2 > <B1> <B0> <mouse> <dom> <tra> <rot>	
<mouse>	This value is ignored.	
<dom>	= 1	All components in the data packet are set to zero except the component with the largest magnitude.
	= 0	The bit has no effect on the components of the data packet.
<tra>	= 1	The bit 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	The bit 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 Magellan/SPACE MOUSE 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 Magellan/SPACE MOUSE 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 Magellan/SPACE MOUSE format. Note that the baud rate does not change; the Magellan/ SPACE MOUSE still sends data at 9600 baud. The maximum data packet transmission rate is 40 ms.
	= 0	Data packets are transmitted in the standard Magellan/SPACE MOUSE format.
Returns:	The selected compressed mode (in the same format as the command).	

Compress Mode ? Command

Function:	Interrogates the current operating mode of the Magellan/SPACE MOUSE.
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 Magellan/SPACE MOUSE.
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 Magellan/SPACE MOUSE is not able to transmit data packets with a time period shorter than 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 Magellan/SPACE MOUSE does not transmit any further data until the cap is moved again.

Sensitivity Setup Command

Function:	Sets the sensitivity of the Magellan/SPACE MOUSE. 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	The relationship is linear.
	= 1...15	A corresponding ballistic (quadratic) function is used.
<sensitivity rot>	= 0	The relationship is linear.
	= 1...15	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:

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

Sensitivity ? Command

Function:	Interrogates the current sensitivity values of the Magellan/SPACE MOUSE.	
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 Magellan/SPACE MOUSE.	
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 ? 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 Magellan/ SPACE MOUSE for the specified amount of time.	
Command:	b<nibble>\r	
<nibble>	<B7...4> < B3 ><B2><B1><B0> <on/off> < duration >	
<on/off>	= 1	Magellan/SPACE MOUSE beeps for the specified duration.
	= 0	The beeper is silent for the 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 Magellan/SPACE MOUSE for the specified amount of time.	
Command:	f<nibble>\r	
<nibble>	<B7...4> < B3 ><B2><B1><B0> < S/R >< duration >	
<S/R>	= 1	The Magellan/SPACE MOUSE flasher is set to a simple flash mode for the specified duration (all LEDs are illuminated simultaneously).
	= 0	The Magellan/SPACE MOUSE flasher is set to a 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.
	= 0	Turns the red LEDs off.
<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 ? 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 Magellan/SPACE MOUSE 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 Magellan/SPACE MOUSE 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 Magellan/SPACE MOUSE 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>\r	
<nibble1>	= 1	Magellan/SPACE MOUSE 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 Magellan/SPACE MOUSE is ready to receive a new command.
	= 2	Magellan/SPACE MOUSE 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 Magellan/SPACE MOUSE 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.

The following hints should be helpful for developing software using Magellan/SPACE MOUSE.

Transmitting the First Command

After receiving the first valid 3D command via the serial port (9600 Baud, 8 data bits, 2 stop bits), the Magellan/SPACE MOUSE 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 Magellan/SPACE MOUSE.

Checking the Handshake Signals

Never transmit more than five bytes to Magellan/SPACE MOUSE without checking the handshake signal (CTS) status. Loss of data and maladjustments of the Magellan/SPACE MOUSE 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 Magellan/SPACE MOUSE. If echo mode is not turned off, more than five characters might be transmitted to the Magellan/SPACE MOUSE without checking the handshake signal status. In addition, Magellan/SPACE MOUSE will erroneously interpret each echo as a command.

Carriage Return Character "\r"

The commands transmitted to Magellan/SPACE MOUSE 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 Magellan/SPACE MOUSE 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 Magellan/SPACE MOUSE 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 Magellan/SPACE MOUSE unless a command without a terminating "\r" character is received. However, reactions to displacements of the Magellan/SPACE MOUSE cap are transmitted only if the translation and rotation are in the ON state.)

Fixed Number of Characters in Commands

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

X-Window Interface

This chapter describes the technical background of the X-Window driver interface, including how data is moved from the X-Window driver to the application. This information may be used to join the Magellan/SPACE MOUSE with the X-Window application. The file *xapp.c*, found on the LogiCad3D CD-ROM under the directory */unix/xdev*, contains a simple example of this procedure.

Software Interface

The exchange of information between the X-Window driver and the application happens in "Events". These are numbers that are globally joined to the X-Window system. These events are exchanged as client messages between the partners. Four different events are joined together.

- *CommandEvent*. A command event.*MotionEvent*. An event of 3D files of movements in X, Y, Z, A, B and C.
- *ButtonPressEvent*. An event of one pressed key.
- *ButtonReleaseEvent*. An event of one released key.

These events are joined together with the help of the *XInternAtom* function and the X-Window system. The numbers returned by *XInternAtom* are exclusively reserved for these events. If the application is in relation to an event (over the *XInternAtom* function), it will receive the same number the X-Window driver received at the installation of the interface. These events equally formulate the joining parts between the X-Window driver and the application. The following program excerpt arranges the agreement of the X-Window System.

```
AtomMotionEventNumber;
AtomButtonPressEventNumber;
AtomButtonReleaseEventNumber;
AtomCommandEventNumber;

MotionEventNumber =
   XinternAtom(display,"MotionEvent",TRUE);
ButtonPressEventNumber =
   XinternAtom(display,"ButtonPressEvent",TRUE);
ButtonReleaseEventNumber =
   XinternAtom(display,"ButtonReleaseEvent",TRUE);
CommandEventNumber =
   XInternAtom(display,"CommandEvent",TRUE);
```

X-Window Driver Window

The X-Window driver opens a window in which it starts the application. Events can only be sent to windows, but they can always be received by the receiver (because the client message information cannot be marked by the receiver). For the application to receive the window number of the X-Window driver, the driver must put down the window number as a characteristic of the *CommandEvent*. From there the application can read the window number of the X-Window driver. The following program excerpt shows how this is possible.

```
AtomActualType;
intActualFormat;
unsigned long NItems,BytesReturn;
unsigned char *PropReturn;
Window root,MagellanWindow;

root = DefaultRootWindow(display);
XGetWindowProperty(display,root,CommandEventNumb
   er,0,1,FALSE,
AnyProperty,&ActualType,&ActualFormat,&NItems,&Byte
   sReturn,&PropReturn);
if ( PropReturn != NULL )
MagellanWindow = *(Window *)PropReturn;
else
MagellanWindow = NULL; /* driver not found */
```

Placing the Application Window

The X-Window driver sends the events it receives from the Magellan/SPACE MOUSE to the current input focus window, which the X-Window system has been associated with through the keyboard input. The driver asks via the *InputFocus* constant which window currently belongs to the input focus. It is also possible to use a *CommandEvent* to let the X-Window driver know which window will receive routed events. This causes the driver to lose the input focus and send all the events without exception to the active window. However, the driver automatically changes back to the input focus when an event cannot be sent successfully to the active window. (This occurs, for example, if the active window has been closed). The following program excerpt shows how to place the window in the X-Window driver.

```
Xevent Event;
Event.type = ClientMessage;
Event.xclient.format = 16;
Event.xclient.send_event = FALSE;
Event.xclient.display = display;
Event.xclient.window = MagellanWindow;
```

```
CommandEventNumber;
Event.xclient.data.s[0] = XHigh32( window );
Event.xclient.data.s[1] = XLow32( window );
Event.xclient.data.s[2] =
    CommandMessageApplicationWindow; /* =1 */
XSendEvent(display,MagellanWindow,FALSE,0x0000,
    &Event);
XFlush(display);
```

The following table shows the structure of *CommandEvent* data.

xclient.data	***16-bit Word***
s[0]	Application Window High 16 bit
s[1]	Application Window Low 16 bit
s[2]	1
s[3]...s[9]	reserved

Receiving Events

When the X-Window driver receives events from the Magellan/SPACE MOUSE, it copies the information into a data structure and sends it to the application as a *ClientMessageEvent*. *MotionEvents*, *ButtonPressEvents* and *ButtonReleaseEvents* may also be sent. The procedure for after the events have been intercepted by the application depends on the particular application. The following program excerpt shows how the events can be intercepted by the application. Note that the reading of the information from the event is displayed in three phases.

```
XEvent Event;

XNextEvent(display,&Event);
switch( Event.report)
  {
  case ClientMessage:
  if ( Event.xclient.message_type ==
      MotionEventNumber )
  /* a 3D motion event is received */;
  if ( Event.xclient.message_type ==
      ButtonPressEventNumber )
  /* a button press event is received */;
  if ( Event.xclient.message_type ==
      ButtonReleaseEventNumber )
  /* a button release event is received */;
  break;
  };
```

3D Movement Event

In this event the X-Window driver sends the application three translational values (X,Y and Z), three rotational values (A,B and C) and the period of duration. Each value requires 16 bits. These values are used to interpret the speed of the graphical object. The normal range of these values is about ±400, but increasing the sensitivity increases the range to approximately ±6000. The following program excerpt reads the movement files out of the data packet.

```
int X,Y,Z,A,B,C,Period;
Window MagellanWindow;

MagellanWindow = (Event.xclient.data.s[0] << 16) |
    Event.xclient.data.s[1];

X = Event.xclient.data.s[2];
Y = Event.xclient.data.s[3];
Z = Event.xclient.data.s[4];

A = Event.xclient.data.s[5];
B = Event.xclient.data.s[6];
C = Event.xclient.data.s[7];

Period = Event.xclient.data.s[8];  /*e.g. 60 means 60ms
    */
```

The following table shows the structure of *MotionEvent* data.

xclient.data	***16-bit Word***
s[0]	Magellan Window High 16 bit
s[1]	Magellan Window Low 16 bit
s[2]	X Translation
s[3]	Y Translation
s[4]	Z Translation
s[5]	A Rotation
s[6]	B Rotation
s[7]	C Rotation
s[8]	Period
s[9]	reserved

Event of a Pressed Key

Whenever a key on the Magellan/SPACE MOUSE is pressed, the X-Window driver sends the number of the pressed key to the application. The numbered keys (1-8) are represented by the corresponding numbers. The star key (*) is represented by 9. The following program excerpt reads the number of the pressed key from the event.

```
int ButtonNumber;
Window MagellanWindow;

MagellanWindow = (Event.xclient.data.s[0] << 16) |
    Event.xclient.data.s[1];
ButtonNumber = Event.xclient.data.s[2];
```

The following table shows the structure of *ButtonPressEvent* data.

xclient.data	16-bit Word
s[0]	Magellan Window High 16 bit
s[1]	Magellan Window Low 16 bit
s[2]	Keyboard Number
s[3]...s[9]	reserved

Event of a Released Key

Whenever a key on the Magellan/SPACE MOUSE is released, the X-Window driver sends the number of the released key to the application. The numbered keys (1-8) are represented by the corresponding numbers. The star key (*) is represented by 9. The following program excerpt reads the number of the released key from the event.

```
int ButtonNumber;
Window MagellanWindow;

MagellanWindow = (Event.xclient.data.s[0]
    <<16) | Event.xclient.data.s[1];
ButonNumber = Event.xclient.data.s[2];

int ButtonNumber;
Window MagellanWindow;

MagellanWindow = (Event.xclient.data.s[0]
    <<16) | Event.xclient.data.s[1];
ButonNumber = Event.xclient.data.s[2];
```

ButtonReleaseEvent data has the same structure as *ButtonPressEvent* (see description the table above).

The *xdrvlib.c* Library

A simpler way to use the Magellan/SPACE MOUSE interface is to use the existing calls in the *xdrvlib.c* library, which is found on the LogiCad3D CD-ROM under the directory */unix/xdev*. Using the library makes the construction of the interface easier, since the application only needs to read data out of the predefined structures. This section explains the functions, a global variable and a defined data structure.

MagellanInit

This function initializes the software interface to the X-Window driver. It combines the four events (*CommandEvent*, *MotionEvent*, *ButtonPressEvent* and *ButtonReleaseEvent*) with the X-Window system. Furthermore, it tests to see if the X-Window driver is active and places the received window *<window>* into the driver. All subsequent Magellan/SPACE MOUSE events are sent to this window.

Syntax:	int MagellanInit (<display>, <window>) Display *<display>; Window <window>;
<display>	Makes the connection to the X-server firm (see *XOpenDisplay* in Xlib Programming Manual).
<window>	The number of an opened window or the constant *InputFocus*, i.e. *PointerWindow*
Result:	If the function succeeds, it returns the value TRUE. If not, it returns the value FALSE.
File:	xdrvlib.h

MagellanSetWindow

This function places the received window *<window>* into the driver. All subsequent events of the Magellan/SPACE MOUSE are sent to the newly activated window.

Syntax:	int MagellanSetWindow (<display>, <window>) Display *<display>; Window <window>;
<display>	Makes the connection to the X-server firm (see *XOpenDisplay* in Xlib Programming Manual).
<window>	The number of an opened window or the constant *InputFocus*, i.e. *PointerWindow*
Result:	If the function succeeds, it returns the value TRUE. If not, it returns the value FALSE.
File:	xdrvlib.h

MagellanTranslateEvent

This function translates the information from the *ClientMessageEvent* into the data structure of the Magellan/SPACE MOUSE. Reading data from this structure is easy because the *MagellanTranslateEvent* modifies only those values that are valid to the served event. However, using *MagellanMotionEvent* means that the 3D data is multiplied by the scaling factors *<MagellanTraScale>* and *<MagellanRotScale>*. The returned value shows which event has been translated.

Syntax:	int MagellanTranslateEvent (<display>,<Event>,<MagellanEvent>, <MagellanTraScale>,<MagellanRotScale>) Display *<display>; XEvent *<Event>; MagellanFloatEvent *<MagellanEvent>; double <MagellanTraScale>; double <MagellanRotScale>;
<display>	Makes the connection to the X-server firm (see *XOpenDisplay* in Xlib Programming Manual).
<Event>	Contains a *ClientMessageEvent*, which is sent from the X-Window driver. It is received with the function *XNextEvent* or a similar function (e.g. *XPeekEvent*, *XWindowEvent*).

<Magellan Event>	A pointer on the data structure in which all information on the served event is stored.
<Magellan TraScale>	The scaling factor multiplied by the 3 translation values of a *MagellanMotion Event*.
<Magellan RotScale>	is the scaling factor multiplied by the 3 rotation values of a *MagellanMotionEvent*.
Result:	The function returns the number of the translated event. The constants *MagellanInputMotion Event*, *MagellanInputButton PressEvent* and *MagellanInput ButtonReleaseEvent* are possible. If the served event is not an event coming from the X-Window driver, it can only return the value FALSE.
File:	xdrvlib.h

The following table describes the structure of the Magellan/SPACE MOUSE data.

Element Type	Element Name	Description
int	*Magellan Type*	Contains the last stored event type. It can take one of the three following values: *MotionEvent* *ButtonPressEvent* *ButtonReleaseEvent* These three values are combined as constants and can be used within switch operations.
int	*Magellan Button*	Contains the number of the last pressed or released Magellan/ SPACE MOUSE key.
double	*Magellan Data*[6]	Contains the three translational and three rotational values.
int	*Magellan Period*	Indicates which repeating data rate of the Magellan/SPACE MOUSE can be used for transmitting. The value is given in milliseconds.

MagellanRotationMatrix

This function calculates the rotation matrix from the 3 rotational inputs. Note that the argument *<Rotate>* is a 4x4 matrix whose first three rows and columns store the 3x3 rotation matrix (see also the related appendix Mathematics of 3D Motion Control).

Syntax:	int MagellanRotationMatrix (<Rotate>, <C>,<B>,<A>) double <Rotate>[4][4]; double <C>,<B>,<A>
<Rotate>	Stores the 4x4 rotation matrix.
<C>	Rotation angle about the z-axis.
<B>	Rotation angle about the y-axis.
<A>	Rotation angle about the x-axis.
Result:	The function always returns the value TRUE.
File:	xdrvlib.h

MagellanMultiplicationMatrix

This function multiplies the data from two rotation matrices (parameters *<MatrixB>* and *<MatrixC>*) and stores the results in *<MatrixA>*. Note that the first three rows and columns of *<MatrixA>* store the 3x3 rotation matrix (see also the related appendix Mathematics of 3D Motion Control).

Syntax:	int MagellanMultiplicationMatrix (<MatrixA>,<MatrixB>,<MatrixC>) double <MatrixA>[4][4]; double <MatrixB>[4][4]; double <MatrixC>[4][4];
<MatrixA>	Stores the result of the multiplication of *<MatrixB>* and *<MatrixC>*.
<MatrixB>	Contains the first operand of the multiplication operation.
<MatrixC>	Contains the second operand of the multiplication operation.
Result:	*The function always returns the value TRUE.*
File:	*xdrvlib.h*

MagellanClose

This function closes the software interface to the X-Window driver. The driver sends the information received from Magellan/SPACE MOUSE events to the window that contains an input focus. The connection to the X-Window driver can be opened at anytime with the function *MagellanInit*.

Syntax:	int MagellanClose (<display>) Display *<display>;
<display>	Makes the connection to the X-server firm (see *XOpenDisplay* in Xlib Programming Manual).
<window>	The number of an opened window or the constant *InputFocus*, i.e. *PointerWindow*
Result:	The function always returns the value TRUE.
File:	xdrvlib.h

MagellanExist

MagellanExist is a global variable that indicates whether the X-Window driver is active. The variable is set as TRUE when the driver is active and as FALSE when it is not. The value of *MagellanExist* can only be changed when called up with the function *MagellanInit* or *MagellanClose*.

Syntax:	int MagellanExist;
File:	xdrvlib.h

The *mgldll.c* library contains several useful functions for constructing the interface for a Windows application. The file is found on the LogiCad3D CD-ROM under the directory *|win95nt|source|mgldrv|*. The following sections describe the functions available in this library.

MagellanInit

This function checks for a running Magellan/ SPACE MOUSE driver and initializes the interface. It must be the first Magellan/SPACE MOUSE function to be called.

Syntax:	HMAGELLAN MagellanInit (/* [in] */ HWND ApplicationWindow);
Application Window	Handle to the application window. Normally, this input parameter is a variable initialized by the Win32 function *InitClassWindow().*
Result:	If the function succeeds, the return value is a Magellan handle. If not, it returns the value NULL.
File:	mgldll.h

MagellanSetWindow

This function sets the window handle (stored in *ApplicationWindow*) into the Magellan driver as the current application window. All messages generated by the Magellan driver are now sent to the given window. Note that this function has an operation similar to the command *MagellanSetWindowCommand* in the function *MagellanInfoWindow*.

Syntax:	int MagellanSetWindow (/* [in] */ HMAGELLAN MagellanHandle, /* [in] */ HWND ApplicationWindow);
Magellan Handle	Magellan handle. Normally, this parameter is a variable initialized by the function *MagellanInit*.
Application Window	Handle to the application window.
Result:	If the function succeeds, it returns the value TRUE. If not, it returns the value FALSE.
File:	mgldll.h

MagellanTranslateEvent

This function identifies and decodes incoming event messages from the Magellan/SPACE MOUSE driver and stores the resulting information into the data structure pointed by *MagellanEvent*. The Magellan driver sends a message to the application window handler if one of three different types of events occurs: *MotionEvent*, *ButtonPressEvent* or *ButtonReleaseEvent*.

Syntax:	int MagellanTranslateEvent (/* [in] */ HMAGELLAN MagellanHandle, /* [in] */ LPMSG Message, /* [out] */ MagellanIntegerEvent *MagellanEvent);
Magellan Handle	Magellan handle. Normally, this parameter is a variable initialized by the function *MagellanInit*.
Message	Points to a standard Win32 MSG structure with the raw data as sent by the driver.

Magellan Event	This output variables stores the decoded information in a structure of type *MagellanIntegerEvent*, defined in the file mgldll.h as follows:
	struct _MagellanIntegerEvent_ {
	HWND MagellanWindow;
	int MagellanType;
	int MagellanButton;
	int MagellanData[6];
	int MagellanPeriod;
	};
	typedef struct _MagellanIntegerEvent_
	MagellanIntegerEvent;

Magellan Window	The handle to the window of the Magellan/SPACE MOUSE driver.
MagellanType	The event type. Can have one of the values *MotionEvent*, *ButtonPressEvent* or *ButtonReleaseEvent*, as defined in *mgldll.h*.
Magellan Button	Stores the number (1 to 12) of the button which was pressed or released.
MagellanData	An integer array with six elements, defined as follows: *Index* *Axis* 0 X 1 Y 2 Z 3 A 4 B 5 C
Magellan Period	Holds the time period of the data packet sent by the device (in milliseconds). Normally, this value is set to 60 ms.
Result:	If the message parsed by the function corresponds to an event sent by the Magellan driver, the function returns the same value as stored in the *MagellanType* element of *MagellanEvent*. If not, it returns the value FALSE.
File:	mgldll.h

MagellanClose

This function closes the interface to the Magellan driver and must be last Magellan function called by the application.

Syntax:	int MagellanClose (/* [in] */ HMAGELLAN MagellanHandle);
Magellan Handle	Magellan handle. Normally, this parameter is a variable initialized by the function *MagellanInit*.
Result:	If the function succeeds, it returns the value TRUE. If not, it returns the value FALSE.
File:	mgldll.h

MagellanInfoWindow

This function commands the Magellan driver to execute several different functions depending on the *MagellanInfoType* parameter.

Syntax:	int MagellanInfoWindow (/* [in] */ HMAGELLAN MagellanHandle, /* [in] */ WPARAM MagellanInfoType, /* [in] */ LPARAM MagellanInfo);
Magellan Handle	Magellan handle. Normally, this parameter is a variable initialized by the function *MagellanInit*.
MagellanInfo Type	Contains the following elements: *MagellanSetWindowCommand* *MagellanApplicationSensitivity* *MagellanRingBell* *MagellanApplicationStarts* *MagellanModeChange* *MagellanNullRadiusChange* *MagellanControlPanel*
MagellanSet Window Command	Set the window handle, stored in *MagellanInfo*, into the Magellan/SPACE MOUSE driver as the current application window. All messages created by the Magellan/SPACE MOUSE driver are now sent to the given window. A similar call from a different client or an application change (focus window) could cause a change of the application window.
Magellan Application Sensitivity	Set the current application sensitivity stored in *MagellanInfo* (type is *float*) into the Magellan/SPACE MOUSE driver. Note that the feature to display the current application sensitivity has been disabled since MGLDRV.EXE Driver Version 4.60.
MagellanRing Bell	Activate the internal beeper of the Magellan/SPACE MOUSE device. The duration (in milliseconds) is given in *MagellanInfo*.
Magellan Application Starts	Ignored.
Magellan ModeChange	Turn on or off the buttons that permit mode changes (translation, rotation and dominant ON/OFF) in the *Panel* window of the Magellan/SPACE MOUSE driver. When *MagellanInfo* is set to 1, mode changes are enabled. When *MagellanInfo* is set to 0, mode changes are disabled.
MagellanNull Radius Change	When *MagellanInfo* is set to 0, the user cannot change the Null Radius using the device keyboard. If set to 1, the user is allowed to change the Null Radius using the device keyboard (*7).
Magellan ControlPanel	Turn on or off the menu entry *Panel* of the Magellan/SPACE MOUSE driver. When *MagellanInfo* is set to 1, the *Panel* menu entry is enabled. When *MagellanInfo* is set to 0, the *Panel* menu entry is disabled.
MagellanInfo	Argument for *MagellanInfoType*. Stores different data types. For more information see *MagellanInfoType*, above.
Result:	If the function succeeds, it returns the value TRUE. If not, it returns the value FALSE.
File:	mgldll.h

Turbo Magellan/SPACE MOUSE

The differences between the standard Magellan/SPACE MOUSE (SSM) and the Turbo Magellan/SPACE MOUSE (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.

Input	*Char.*	*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 Magellan/SPACE MOUSE cap, eight new points must be generated using the eight old points. To accomplish this, the Magellan/SPACE MOUSE 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 Magellan/SPACE MOUSE 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 Magellan/SPACE MOUSE 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 Magellan/SPACE MOUSE, 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 Magellan/SPACE MOUSE 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 Magellan/SPACE MOUSE has now sent the following second set of values to the computer:

X = 1.5	Y = 0	Z = 0
A = 0	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 Magellan/SPACE MOUSE continues to send values, these computational steps must be repeated recursively.

Product Specifications

Feature/Specification	*Magellan/SPACE MOUSE Classic*	*Magellan/SPACE MOUSE Plus*	*Magellan/SPACE MOUSE Plus XT*
Contactless, wearless, high-linear measuring system	Yes	Yes	Yes
Operating speed levels (increments of resolution)	600	600	600
Number of freely programmable buttons	9	11	11
Software-controllable keyboard LEDs	No	No	Yes (2 yellow, 1 red)
Quicktip virtual button	Yes	Yes	Yes
Device weight (for stability)	0.665 kg	0.680 kg	0.680 kg
Min. releasing force of the measuring system	0.2 N	0.2 N	0.2 N
Max. user force of the measuring system	4.4 N	4.4 N	4.4 N
Ratio of device weight to min. releasing force	33.2	33.2	33.2
Ratio of device weight to max. user force	1.5	1.5	1.5
Device weight deficit with respect to max. user force	0 %	0 %	0 %
Counter force to compensate max. user force	0 N	0 N	0 N
Min. releasing torque of the device	4 N mm	4 N mm	4 N mm
Max. user torque of the device	100 N mm	100 N mm	100 N mm
Customization of user force	Possible	Possible	Possible
Customizable keyboard template	Yes	Yes	Yes
Operating humidity (non-condensing)	10 to 98% RH	10 to 98% RH	10 to 98% RH
Operating temperature	+5 ... +60 Celsius	+5 ... +60 Celsius	+5 ... +60 Celsius
Storage humidity	10 to 98% RH	10 to 98% RH	10 to 98% RH
Storage temperature	-40 ... +85 Celsius	-40 ... +85 Celsius	-40 ... +85 Celsius
Gravity height of work center	25 mm	26 mm	26 mm
Dominant Mode	Yes	Yes	Yes
Supported operating systems	UNIX: DEC, HP, IBM, SGI, SUN PC: Win95/98, WinNT/2000, DECNT, MIPSNT	UNIX: DEC, HP, IBM, SGI, SUN PC: Win95/98, WinNT/2000, DECNT, MIPSNT	UNIX: DEC, HP, IBM, SGI, SUN PC: Win95/98, WinNT/2000, DECNT, MIPSNT
Power source	5V / 9mA	5V / 9mA	5V / 9mA
Interface type	RS232, 9600 Baud	RS232, 9600 Baud (optional 19k Baud)	RS232, 9600 Baud (optional 19k Baud)
Internal resolution	8 bit	8 bit	8 bit
Cable length	2 m	2 m (optional 3m)	2 m (optional 3m)
Connector	9-p D-Sub f	9-p D-Sub f	9-p D-Sub f
Converter-adapters available for the following RS232 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
Drift-free with temperature change	Yes	Yes	Yes
Resistant to aging effects (constant values measured)	Yes	Yes	Yes
Standard data rate	40 ms	40 ms (opt. 18 ms)	40 ms (opt. 18 ms)
Dust and splash water protection	Possible	Possible	Possible
FCC, TUV/GS, UL/UR, CE Approved	Yes	Yes	Yes
Length of manufacturer's warranty	3 years	3 years	3 years
Standard driver source freely available	Yes	Yes	Yes
Compact size L x W x H	165 x 112 x 40 mm	188 x 120 x 44 mm	188 x 120 x 44 mm

References

Holmes, Michael and Bob Flanders. *C++ Communications Utilities*. PC Magazine. ISBN 1-56276-110-2.

Nye, Adrian. *Xlib Reference Manual for X11 Release 4 and Release 5*, 3rd ed. Volume One. ©1997 O'Reilly & Associates, Inc. ISBN 1-56592-002-3.

Nye, Adrian. *Xlib Reference Manual for X11 Release 4 and Release 5*, 3rd ed. Volume Two. ©1997 O'Reilly & Associates, Inc. ISBN 1-56592-006-6.

The X Resource Issue Four. O'Reilly & Associates, Inc. ISBN 0-937175-99-4.

LogiCad3D Support

If you have any questions or comments about the Magellan/ SPACE MOUSE product, please contact the persons or organizations listed below for your area. Various information about the Magellan/ SPACE MOUSE, including the latest driver versions, can be found at the web sites.

America & Asia

LogiCad3D, Inc.
17672 Laurel Park Drive North, Suite 400
Livonia, MI 48152
U.S.A.
Tel: + 1-734-591-4047
Fax: + 1-734-591-4064
Email: logicad3d.US@logicad3d.com
Web: www.logicad3d.com

Marketing & Sales
Miguel Leitmann
Tel: + 1-734-591-4047
Email: Miguel.Leitmann@logicad3d.com

Support
Tel: + 1-800-540-4758
Email: support.US@logicad3d.com

Help
Email: help@logicad3d.com

EMEA (Europe, Middle East & Africa)

LogiCad3D GmbH
An der Hartmuehle 8
D-82229 Seefeld
Germany
Tel: + 49 (0) 8152-9919-0
Fax: + 49 (0) 8152-9919-50
Email: logicad3d.EU@logicad3d.com
Web: www.spacemouse.com

Marketing & Sales
Ralf Stetter
Tel: + 49 (0) 8152-9919-0
Email: Ralf.Stetter@logicad3d.com

Support
Tel: + 49 (0) 8152-9919-44
Email: support.EU@logicad3d.com

Help
Email: help@logicad3d.com

Warranty Information

LogiCad3D's Limited Lifetime Hardware Warranty

LogiCad3D warrants that Magellan/SPACE MOUSE is free from significant defects in materials and workmanship under normal use for as long as the device is owned. During the first three years of ownership, LogiCad3D will at its sole option, replace or repair at no charge the product, which in its opinion is defective.

During the remaining years of the warranty, LogiCad3D will, at its sole option, replace or repair the defective product. LogiCad3D will charge a fixed fee to cover handling and service costs based on LogiCad3D's then current price schedules. LogiCad3D at its sole option, may replace or repair the defective product with a then current product having similar features and functionality as determined by LogiCad3D.

Damages or defects to the product caused by improper installation, modification, misuse or abuse are not, of course, covered by this warranty. Additionally, the warranty service offered above is nontransferable, which means that the particular warranty service described above is available only to the original purchaser of the product(s). If LogiCad3D determines that the product is not defective or was not under warranty, it will return the product(s) to you, freight collect.

To obtain warranty service, you must either (a) have a completed warranty registration card on file at LogiCad3D, or (b) submit acceptable proof of purchase (for instance, a copy of your sales receipt indicating date and place of purchase) to LogiCad3D's Customer Service Department. You will need to pack the product to be returned properly for shipment and pay any applicable shipping charges. LogiCad3D will send you the repaired or replaced product at its own expense.

LOGICAD3D DISCLAIMS ALL OTHER WARRANTIES, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE WITH REGARD TO THE PRODUCT. IN NO EVENT SHALL LOGICAD3D OR ITS SUPPLIERS BE HELD LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL OR INCIDENTAL DAMAGES ARISING OUT OF THE USE OF OR INABILITY TO USE THE PRODUCT, EVEN IF LOGICAD3D HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF LIABILITY FOR CONSEQUENTIAL OR INCIDENTAL DAMAGES, SO THE ABOVE LIMITATION MAY NOT APPLY TO YOU.

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 Magellan/SPACE MOUSE 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.

LogiCad3D Software License Agreement

This is a legal agreement between you (either an individual or entity), the end user License ("Licensee") and LogiCad3D Inc. ("LogiCad3D"). If you do not agree to the terms of this Agreement, promptly return the disks and the other items that are a part of this product in their original package with a copy of your payment receipt for a full refund.

No part of this Software and all accompanying documentation, including manuals, binders and containers (the "Software") may be copied or reproduced in any form or by any means without the prior written consent of LogiCad3D Inc. with the one exception that the Licensee may copy the Software solely for backup purposes.

License Grant

LogiCad3D grants to the Licensee a nonexclusive right, without right to sublicense, to use this copy of Software on a single computer at a time. You may not rent or lease the Software, but you may transfer the Software on a permanent basis, provided you retain no copies and the recipient agrees to the terms of this Agreement. You may not reverse- engineer, decompile or disassemble the Software. Further, you may not network the Software or otherwise use it on more than one computer or computer terminal at the same time. The Software is owned by LogiCad3D or its suppliers and is protected by United States copyright laws and international treaty provisions.

Limited Warranty

LogiCad3D warrants that (a) the Software will perform substantially in accordance with the accompanying written materials for a period of (90) days from the date of receipt and (b) any hardware accompanying the Software will be free from defects in materials and workmanship under normal use and service for a period of three years from the date of receipt. Any implied warranties on the Software and hardware are limited to 90 days and three (3) years, respectively. Some states do not allow limitations on duration of an implied warranty, so the above limitation may not apply to you.

End User Remedies

LogiCad3D's entire liability and your exclusive remedy shall be for any breach of warranty, at LogiCad3D's option, either (a) return of the price paid or (b) repair or replacement of the Software or hardware that does not meet LogiCad3D's Limited Warranty; provided that the Software and hardware must be returned either to LogiCad3D or to the point of purchase with a copy of your receipt. This Limited Warranty is void if failure of the Software and or hardware has resulted from accident, abuse or misapplication. Any replacement Software or hardware will be warranted for the remainder of the original warranty period or 30 days, whichever is longer.

No Other Warranties

LOGICAD3D DOES NOT WARRANT THE SOFTWARE IS ERROR-FREE. LOGICAD3D DISCLAIMS ALL OTHER WARRANTIES, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD-PARTY RIGHTS WITH RESPECT TO THE SOFTWARE OR HARDWARE. LOGICAD3D'S LIMITED WARRANTY GIVES YOU SPECIFIC LEGAL

RIGHTS. YOU MAY HAVE OTHERS, WHICH VARY FROM STATE TO STATE.

Limitation of Liabilities

IN NO EVENT SHALL LOGICAD3D OR ITS SUPPLIERS BE LIABLE FOR ANY CONSEQUENTIAL, INCIDENTAL OR INDIRECT DAMAGES OF ANY KIND WHATSOEVER (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION OR OTHER PECUNIARY LOSS) ARISING OUT OF THE USE OF OR INABILITY TO USE THIS LOGICAD3D PRODUCT, EVEN IF LOGICAD3D HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN NO EVENT SHALL LOGICAD3D'S LIABILITY EXCEED THE LICENSE FEE PAID. BECAUSE SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF LIABILITY FOR CONSEQUENTIAL OR INCIDENTAL DAMAGES, THE ABOVE LIMITATIONS MAY NOT APPLY TO YOU.

Terms

This license is effective until terminated. You may terminate it at any time by destroying the Software. It will also terminate upon conditions set forth elsewhere in this Agreement or if you fail to comply with any terms or conditions of this Agreement. You agree upon such termination to destroy the Software together with all copies, modifications and merged portions in any form.

General

This is the entire agreement between you and LogiCad3D, superseding any prior agreement whether written or oral relating to the subject matter of this Agreement. In the event of invalidity of any provision of this Agreement, the parties agree that such invalidity shall not affect the validity of the remaining potions of the Agreement. This Agreement will be governed by the laws of the state of California. The United Nations Convention on Contracts for the International Sale of Goods is specifically disclaimed.