

UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE QUITO – CAMPUS SUR

CARRERA DE INGENIERÍA DE SISTEMAS

MENCIÓN ROBÓTICA E INTELIGENCIA ARTIFICIAL

**DESARROLLO DE UN “GUANTE ELECTRÓNICO DE DATOS”
CON SENSORES INERCIALES, HERRAMIENTAS OPEN SOURCE
Y COMUNICACIÓN INALÁMBRICA, QUE INTERACTÚE CON
IMÁGENES DEL CUERPO HUMANO EN 3D PARA EL PROYECTO
“SISTEMA DE ENTRENAMIENTO VIRTUAL PARA MEDICINA”**

TESIS PREVIA A LA OBTENCIÓN DEL TÍTULO DE INGENIERO DE SISTEMAS

**PILLAJO OBANDO MARCELO FERNANDO
ROBAYO CAJAMARCA SIXTO BOLÍVAR**

**DIRECTOR
ING. WASHINGTON RAMÍREZ
QUITO, NOVIEMBRE 2012**

DECLARACIÓN

Nosotros, Marcelo Fernando Pillajo Obando y Sixto Bolívar Robayo Cajamarca, declaramos bajo juramento que el trabajo aquí descrito es de nuestra autoría; que no ha sido previamente presentada para ningún grado o calificación profesional; y que hemos consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedemos nuestros derechos de propiedad intelectual correspondiente a este trabajo, a la Universidad Politécnica Salesiana, según lo establecido por la Ley de Propiedad Intelectual, por su reglamento y por la normativa institucional vigente.

Marcelo Fernando Pillajo Obando

Sixto Bolívar Robayo Cajamarca

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por Marcelo Fernando Pillajo Obando y Sixto Bolívar Robayo Cajamarca bajo mi dirección.

ING. WASHINGTON RAMÍREZ

Director de tesis

DEDICATORIA

Este trabajo esta dedicado a todas las personas, amigos y hermanos, que estuvieron para apoyarme en los tiempos difíciles y también en tiempos buenos, de manera muy especial a mi madre, que con su vida me mostró que hay que esforzarse cada día sin importar las limitaciones y que mañana, siempre será mejor que hoy.

Marcelo Fernando Pillajo Obando.

A Dios.

Por permitirme llegar hasta este momento, brindándome salud y fortaleza logrando así cristalizar mis metas propuestas.

A mis padres.

Porque fueron el pilar fundamental durante mi vida universitaria y el desarrollo de la tesis, ya que siempre estuvieron apoyándome en todo momento entregándome ejemplos dignos de superación y entrega.

A mis amigos.

Por esas palabras de ánimo que han fomentado el deseo de seguir adelante incluso en los momentos más difíciles haciendo de ésta, una experiencia maravillosa.

A mi director de tesis.

Por ser una gran persona, un amigo y brindar su apoyo incondicional en todo momento en esta fase de culminación de mis estudios profesionales y la elaboración de esta tesis, Msc. Washington Ramírez.

Sixto Bolívar Robayo Cajamarca.

AGRADECIMIENTOS

Gracias te damos Dios por amarnos, fortalecernos, sustentarnos y poner la sabiduría necesaria que nos permitió avanzar en cada etapa del desarrollo y culminación de este trabajo.

La realización de este proyecto ha sido también gracias a la colaboración de las siguientes personas:

A nuestros familiares, por su apoyo incondicional.

A nuestro director de tesis Ingeniero Washington Ramírez por permitirnos ser parte del proyecto que el dirigía.

A la Ingeniera Doris Meza por su aporte, que ha sido de mucha ayuda para nosotros.

A nuestros amigos por el ánimo que recibimos en todo momento.

Marcelo Fernando Pillajo Obando.

Sixto Bolivar Robayo Cajamarca.

RESUMEN

Este trabajo de investigación y desarrollo tecnológico presenta como resultado, la fusión de dos componentes como son: el hardware y el software, reunidos en el denominado "Guante Electrónico de Datos" empleado como periférico de entrada para trabajar en un espacio tridimensional; de esta manera su uso no se limita a la operación sobre una superficie plana, sino que logra emular el funcionamiento de un mouse aéreo para el computador por medio de sensores inerciales: giroscopios y acelerómetros que permite localizar su posición. Para la interacción del usuario con el computador se utilizan sensores de contacto colocados en partes estratégicas de los dedos del guante, esta interacción se logra mediante comunicación inalámbrica a través del protocolo Zig Bee. La investigación está basada en herramientas Open Source Hardware y Software mediante el uso de la plataforma Arduino y Ubuntu. Cabe destacar que este trabajo busca explotar la navegación inercial a través del uso de herramientas libres para hardware, cuyo uso no está aún conocido en el medio local. A través del método analítico en donde se evaluaron diversas posibilidades de combinación de elementos, y desarrollo de prototipos en donde se utilizaron los elementos seleccionados, se ha logrado generar un modelo basado en los objetivos planteados al inicio del proyecto; esto no implica que grupos interesados puedan utilizarlo como base para nuevos estudios.

ABSTRACT

This research work and technological development presents as result, the fusion of two components such as: the hardware and software, integrated in a system called "Guante Electrónico de Datos" that used as input peripheral device to work in a three dimensional space. Thus its use is not limited to operation on a flat surface, achieving emulate the operation of an air mouse for the computer, through inertial sensors: gyroscopes and accelerometers that allow locate its position. In order to get a user

interaction with the computer, is using touch sensors placed at strategic parts of the glove fingers, this interaction is achieved by wireless communication through the Zig Bee protocol. The research is based on Open Source Hardware tools and Software using the Arduino platform and Ubuntu Linux. Note that this work seeks to exploit the inertial navigation through the use of free hardware tools whose use is no yet diffused in the local environment. Through the analytical method was evaluated various possible combinations of elements, and prototypes development where were used the selected items, has been achieved to generate a model based on the objectives established at the beginning of the project; this does not imply that interested groups can use as a basis for new studies.

ÍNDICE

1.	INTRODUCCIÓN	1
1.1	PROYECTO	1
1.1.1	PLANTEAMIENTO DEL PROBLEMA	1
1.1.2	OBJETIVOS	2
1.1.3	JUSTIFICACIÓN DEL PROYECTO	3
1.1.4	ALCANCE DEL PROYECTO	4
1.1.5	METODOLOGÍA	5
1.2	OPEN SOURCE	9
1.2.1	OPEN SOURCE SOFTWARE (FOSS)	9
1.2.2	OPEN SOURCE HARDWARE (OSHW)	9
1.3	REALIDAD VIRTUAL	10
1.3.1	DEFINICIÓN	11
1.3.2	HISTORIA	11
1.3.3	CLASIFICACIÓN	13
1.3.4	CARACTERÍSTICAS PRINCIPALES DE LA REALIDAD VIRTUAL	13
1.3.5	DISPOSITIVOS	15
1.3.6	SISTEMA DE REALIDAD VIRTUAL	16
1.3.7	APLICACIONES	16
1.4	GUANTE ELECTRÓNICO DE DATOS	17
1.4.1	VERSIONES ACTUALES	17
1.5	ANATOMÍA DE LA MANO HUMANA	20
1.5.1	EL HOMBRO	20
1.5.2	EI CODO	23
1.5.3	LA MUÑECA	24
1.5.4	LA MANO	26
1.5.5	EL PULGAR	30
1.5.6	TOPOGRAFÍA FUNCIONAL	33
1.5.7	FUNCIÓN PRENSIL	34
1.5.7.1	AGARRE DE FUERZA	34
1.5.8	GRADOS DE LIBERTAD	35
2.	FUNDAMENTOS TEÓRICOS	37
2.1	PRINCIPIO DE INERCIA	37
2.2	SISTEMA DE NAVEGACIÓN INERCIAL (INS)	38

2.2.1	COMPONENTES BÁSICOS DE UN INS	39
2.2.2	CONCEPTOS BÁSICOS DE NAVEGACIÓN INERCIAL	40
2.2.3	SISTEMA DE COORDENADAS	41
2.2.4	TIPOS DE SISTEMA DE NAVEGACIÓN INERCIAL	42
2.2.5	ALGORITMOS DE NAVEGACIÓN	44
2.3	CUATERNIONES	47
2.3.1	OPERACIONES CON CUATERNIONES	48
2.3.2	REPRESENTACIÓN DE PUNTOS A TRAVÉS DE CUATERNIONES	49
2.3.3	REPRESENTACIÓN DE VECTORES A TRAVÉS DE CUATERNIONES	49
2.3.4	CUATERNIONES UNITARIOS	50
2.3.5	GIMBAL LOCK	50
2.3.6	ROTACIONES	51
2.3.7	APLICACIONES	54
2.3.8	VENTAJAS	54
2.3.9	RENDIMIENTO	55
2.4	TECNOLOGÍA MEMS (SISTEMAS MICRO-ELECTRO-MECÁNICOS)	55
2.4.1	TIPOS DE MEMS	56
2.4.2	COMPONENTES MEMS	56
2.4.3	CONSTRUCCIÓN DE LOS MEMS	57
2.4.4	VENTAJAS	58
2.4.5	APLICACIONES	58
2.5	DISPOSITIVOS INERCIALES	58
2.5.1	EL ACELERÓMETRO	58
2.5.2	EL GIROSCOPIO	72
2.6	REGISTRO DE LA ORIENTACIÓN	81
2.6.1	EL ACELERÓMETRO Y LA INCLINACIÓN	81
2.6.2	EL GIROSCOPIO Y LA INCLINACIÓN	84
2.6.3	CÁLCULO DE LA INCLINACIÓN USANDO GIRÓSCOPO Y ACELERÓMETRO	86
2.6.4	FILTRO BASADO EN GRADIENTE DESCENDENTE	87
2.7	PROTOCOLOS DE COMUNICACIÓN	88
2.7.1	PROTOCOLO DE COMUNICACIÓN RS232	88
2.7.2	PROTOCOLO I2C	91
2.7.3	PROTOCOLO ZIGBEE	98
2.8	ARDUINO	112
2.8.1	CARACTERÍSTICAS	112
2.8.2	HARDWARE ARDUINO	113
2.8.3	SOFTWARE ARDUINO	115

3.	ENTORNO DE DESARROLLO Y MATERIALES	119
3.1	HARDWARE	119
3.1.1	ARDUINO NANO USB MICROCONTROLLER V3.	119
3.1.2	DIGI XBee	123
3.1.3	XBee Explorer Dongle	125
3.1.4	MICROCONVERSION USB-serial CP2102	126
3.1.5	IMU 6 GRADOS DE LIBERTAD ADXL345/ITG3200	127
3.1.6	MPU-6050 (UNIDAD DE PROCESAMIENTO DEL MOVIMIENTO)	132
3.2	SOFTWARE	140
3.2.1	INSTALACIÓN DE ARDUINO IDE EN UBUNTU LINUX	140
3.2.2	PUERTO SERIAL EN LINUX	143
3.2.3	INTERACCIÓN GUANTE DE DATOS - PC	148
3.2.4	X-CTU SOFTWARE	156
3.2.5	QT	164
3.3	MATERIALES PARA LA CONSTRUCCIÓN DEL "GUANTE ELECTRÓNICO DE DATOS"	168
3.3.1	GUANTE ONE POLAR.	168
3.3.2	CONDUCTIVE THREAD - 234/34 4ply	168
3.3.3	CONDUCTIVE FABRIC - 12"x13" MedTex130	169
3.3.4	SNAP ASSORTMENT - (MALE AND FEMALE) AND NEEDLE	170
3.3.5	WIRE WRAPPING WIRE	170
3.3.6	LITHIUM POLYMER BATTERY CELL - 3.7V 1000mAh.	171
3.3.7	USB LiPoly Charger - Single Cell	172
3.3.8	VIBRATION MOTOR	172
3.3.9	1N4007 SMD	173
3.3.10	ZLDO1117 SMD	173
3.3.11	TRANSISTOR NPN 2N3904 SMD	173
4.	DESARROLLO DEL GUANTE ELECTRÓNICO DE DATOS	174
4.1	ARQUITECTURA DEL SISTEMA	174
4.2	PRUEBAS INICIALES	175
4.2.1	PRUEBA UNO	175
4.2.2	PRUEBA DOS	178
4.3	PRODUCTO FINAL	181
4.3.1	ESQUEMÁTICO	182
4.3.2	CIRCUITO FINAL	182
4.3.3	GUANTE ELECTRÓNICO DE DATOS PRODUCTO FINAL.	183
4.3.4	PRUEBAS DE OPERACIÓN	197

4.3.5	<i>COSTOS DEL PRODUCTO FINAL</i>	209
5.	CONCLUSIONES Y RECOMENDACIONES	210
5.1	CONCLUSIONES	210
5.1.1	<i>CAPÍTULO UNO</i>	210
5.1.2	<i>CAPÍTULO DOS</i>	211
5.1.3	<i>CAPÍTULO TRES</i>	212
5.1.4	<i>CAPÍTULO CUATRO</i>	213
5.2	RECOMENDACIONES	214
6.	ANEXOS	216
6.1	ANEXO A	216
6.1.1	<i>puerto.h</i>	216
6.1.2	<i>puerto.cpp</i>	217
6.2	ANEXO B	223
6.2.1	<i>xuti.h</i>	223
6.2.2	<i>xutil.cpp</i>	224
6.3	ANEXO C	227
7.	BIBLIOGRAFÍA	240

ÍNDICE DE FIGURAS

Cap.1: Fig. 1	Fases del proyecto de desarrollo tecnológico.	8
Cap.1: Fig. 2	Simulador de realidad virtual inmersivo.	14
Cap.1: Fig. 3	Modelo genérico de un Sistema de Realidad Virtual.	16
Cap.1: Fig. 4	DG5 VHAND 2.0.	17
Cap.1: Fig. 5	DATA GLOVE 5DT.	18
Cap.1: Fig. 6	5DT DATA GLOVE 5 MRI.	18
Cap.1: Fig. 7	CYBER GLOVE II	19
Cap.1: Fig. 8	CYBERGRASP EXOSKELETON.	19
Cap.1: Fig. 9	ACCELEGLOVE	20
Cap.1: Fig. 10	Principales ejes de movimiento del hombro.	21
Cap.1: Fig. 11	Movimiento de abducción y aducción del hombro.	21
Cap.1: Fig. 12	Movimiento de flexión y extensión del hombro.	22
Cap.1: Fig. 13	Movimiento de rotación del hombro derecho.	23
Cap.1: Fig. 14	Flexión y extensión del codo.	24
Cap.1: Fig. 15	Movimientos del codo.	24
Cap.1: Fig. 16	Movimientos de abducción y aducción de la muñeca.	25
Cap.1: Fig. 17	Movimiento Flexión y Extensión de la muñeca.	25
Cap.1: Fig. 18	Movimiento de circunducción de la muñeca.	26
Cap.1: Fig. 19	a) Cara palmar y b) Cara dorsal.	27
Cap.1: Fig. 20	Topografía de la mano.	27
Cap.1: Fig. 21	Movimiento de Metacarpofalange.	29
Cap.1: Fig. 22	Extensión de la mano.	29
Cap.1: Fig. 23	Abducción del dedo índice.	29
Cap.1: Fig. 24	Circunducción del dedo índice.	30
Cap.1: Fig. 25	Anteposición del pulgar.	30
Cap.1: Fig. 26	Flexión del pulgar.	31
Cap.1: Fig. 27	Pronación del pulgar.	31
Cap.1: Fig. 28	Oposición pulgar-meñique.	32
Cap.1: Fig. 29	Movimiento de contraposición.	32
Cap.1: Fig. 30	Nervios motores que accionan la oposición y contraposición.	32
Cap.1: Fig. 31	Total oposición del pulgar.	33
Cap.1: Fig. 32	Funcionamiento del pulgar con todos los dedos de la mano.	33
Cap.1: Fig. 33	Tipos de agarre de la mano.	34
Cap.1: Fig. 34	Oposición terminal.	34

Cap.1: Fig. 35	Oposición subterminal.	35
Cap.1: Fig. 36	Oposición subterminolateral.	35
Cap.1: Fig. 37	Ejemplo de una mano con 18 grados de libertad	36
Cap.2: Fig. 1	Sistema de referencia inercial.	38
Cap.2: Fig. 2	Ejes de movimiento de navegación inercial.	42
Cap.2: Fig. 3	Componentes principales presentes en una plataforma inercial.	43
Cap.2: Fig. 4	Sistema strap-down o plataforma analítica	43
Cap.2: Fig. 5	Sistema de referencia global (X, Y, Z) y móvil (X'', Y'', Z'').	44
Cap.2: Fig. 6	Movimientos de un móvil en ángulos de Euler.	45
Cap.2: Fig. 7	Giro en Yaw	46
Cap.2: Fig. 8	Giro Pitch	46
Cap.2: Fig. 9	Giro Roll	46
Cap.2: Fig. 10	Modelo básico de una plataforma con gimbals	50
Cap.2: Fig. 11	Ejemplo de gimbal look	51
Cap.2: Fig. 12	Demostración de rotación con cuaterniones.	53
Cap.2: Fig. 13	Funcionamiento general de los MEMS.	57
Cap.2: Fig. 14	Proceso general de fabricación de los MEMS	57
Cap.2: Fig. 15	Funcionamiento General de un acelerómetro.	59
Cap.2: Fig. 16	Aceleraciones externas y efectos de la gravedad.	60
Cap.2: Fig. 17	Acelerómetro mecánico.	63
Cap.2: Fig. 18	Acelerómetro piezoresistivo.	63
Cap.2: Fig. 19	Acelerómetro piezoeléctrico.	64
Cap.2: Fig. 20	Acelerómetro capacitivo.	64
Cap.2: Fig. 21	Esquema interno básico de un sensor tipo MEMS.	65
Cap.2: Fig. 22	Detalle de un típico acelerómetro MEMS.	66
Cap.2: Fig. 23	Masa sísmica de un acelerómetro MEMS.	67
Cap.2: Fig. 24	Funcionamiento de un giroscopio.	73
Cap.2: Fig. 25	Giroscopio mecánico.	74
Cap.2: Fig. 26	Esquemático de la estructura vibratoria de un giroscopio MEMS	75
Cap.2: Fig. 27	Velocidades angulares que mide un giroscopio.	76
Cap.2: Fig. 28	Fuerzas que actúan en el efecto Coriolis.	77
Cap.2: Fig. 29	Aceleración Coriolis con respecto a un observador.	78
Cap.2: Fig. 30	Modelo simplificado para un giróscopo de estructura vibratoria.	78
Cap.2: Fig. 31	Efecto Coriolis.	79
Cap.2: Fig. 32	Velocidad angular aplicada.	80

Cap.2: Fig. 33	Funcionamiento interno del giroscopio MEMS.	80
Cap.2: Fig. 34	Calculo de la inclinación usando un sólo eje del acelerómetro.	82
Cap.2: Fig. 35	Cálculo de la inclinación empleando un acelerómetro de 3 ejes.	83
Cap.2: Fig. 36	Implementación de un filtro de orientación para una IMU (Giróscopo + Acelerómetro).	88
Cap.2: Fig. 37	Pines más importantes del conector DB9.	89
Cap.2: Fig. 38	Niveles de tensión para unos y ceros lógicos	89
Cap.2: Fig. 39	Formato de un byte de dato en el Estándar RS-232.	90
Cap.2: Fig. 40	Conversores USB-SERIAL.	91
Cap.2: Fig. 41	Conexión I2C.	92
Cap.2: Fig. 42	Topología I2C.	93
Cap.2: Fig. 43	AND Cableada (a) una o mas salidas a '0' (b) todas las salidas a '1'.	93
Cap.2: Fig. 44	Condición Start y Stop.	94
Cap.2: Fig. 45	Transferencia de datos sobre el protocolo I2C.	95
Cap.2: Fig. 46	Conexión I2C de varios dispositivos con dirección propia.	97
Cap.2: Fig. 47	Logo ZegBee Alliance.	98
Cap.2: Fig. 48	Arquitectura ZigBee.	101
Cap.2: Fig. 49	Estructura de canales de operación Zigbee.	102
Cap.2: Fig. 50	Campos de los cuatro tipos de paquetes básicos de ZigBee.	104
Cap.2: Fig. 51	Topologías Estrella, Árbol y Malla de una red Zigbee.	109
Cap.2: Fig. 52	Aplicaciones Zigbee.	111
Cap.2: Fig. 53	Entorno de programación Arduino.	116
Cap.2: Fig. 54	Programa parpadeo de un led.	118
Cap.3: Fig. 1	ARDUINO NANO USB MICROCONTROLLER V3.	120
Cap.3: Fig. 2	FTDI ubicado en el lado inferior del Arduino Nano V3.	121
Cap.3: Fig. 3	DIGI 1mw XBee.	123
Cap.3: Fig. 4	Designacion pines Módulo XBee.	124
Cap.3: Fig. 5	XBee Explorer Dongle.	126
Cap.3: Fig. 6	Microconversor USB-serial CP2102.	127
Cap.3: Fig. 7	IMU 6 grados de libertad.	127
Cap.3: Fig. 8	Diagrama de bloques y designación de pines ADXL345.	128
Cap.3: Fig. 9	Pines del giroscopio ITG-3200.	130
Cap.3: Fig. 10	Diagrama funcional de bloques ITG-3200.	131
Cap.3: Fig. 11	MPU6050.	132
Cap.3: Fig. 12	Distribución de pines y polarización MPU-6050.	135

Cap.3: Fig. 13	Diagrama de bloques MPU-6050.	135
Cap.3: Fig. 14	Selección del paquete openjdk-7-jre y sus dependencias	141
Cap.3: Fig. 15	Dependencias del paquete openjdk-7-jre.	141
Cap.3: Fig. 16	Selección del compilador gcc-avr.	142
Cap.3: Fig. 17	Selección de las librerías “avr-libc”.	142
Cap.3: Fig. 18	Descarga e instalación de paquetes seleccionados en Ubuntu 12.04.	143
Cap.3: Fig. 19	Arquitectura X11.	149
Cap.3: Fig. 20	Relación Cliente-XLib-Servidor.	150
Cap.3: Fig. 21	Interfaz de usuario X-CTU.	156
Cap.3: Fig. 22	Pestaña PC Settings y sus parámetros de configuración .	158
Cap.3: Fig. 23	Pestaña Range Test.	158
Cap.3: Fig. 24	Pestaña terminal.	159
Cap.3: Fig. 25	Pestaña Modem Configuration.	160
Cap.3: Fig. 26	X-CTU reconoce y visualiza automáticamente los puertos disponibles en la PC.	161
Cap.3: Fig. 27	Detalles de testeo si el modem esta en buenas condiciones y listo para usar.	161
Cap.3: Fig. 28	Parámetros configurables para un Xbee serie 1.	162
Cap.3: Fig. 29	Configuración de direcciones para dos módulos Xbee.	163
Cap.3: Fig. 30	Testeo de la comunicación entre los módulos.	164
Cap.3: Fig. 31	Entorno Qt Creator.	166
Cap.3: Fig. 32	Ejemplo básico con Qt.	167
Cap.3: Fig. 33	Guante mano derecha Onepolar.	168
Cap.3: Fig. 34	Hilo conductor.	169
Cap.3: Fig. 35	Tela conductora.	170
Cap.3: Fig. 36	Corchetes macho y hembra.	170
Cap.3: Fig. 37	Cable aislado.	170
Cap.3: Fig. 38	Batería polímero de litio.	171
Cap.3: Fig. 39	Cargador de baterías LIPO.	172
Cap.3: Fig. 40	Micromotor Vibrador.	172
Cap.3: Fig. 41	Diodo semiconductor 1N4007.	173
Cap.3: Fig. 42	Regulador ZLDO1117.	173
Cap.3: Fig. 43	2N3904.	173

Cap.4: Fig. 1	Arquitectura del sistema.	174
Cap.4: Fig. 2	Esquemático Prueba Uno.	176
Cap.4: Fig. 3	Primer circuito en protoboard.	176
Cap.4: Fig. 4	Esquemático Prueba Dos.	178
Cap.4: Fig. 5	Circuito usando IMU 6 grados de libertad.	179
Cap.4: Fig. 6	Esquemático circuito final.	182
Cap.4: Fig. 7	Circuito final “Guante Electrónico de Datos”.	182
Cap.4: Fig. 8	Guante con 8 grados de libertad.	184
Cap.4: Fig. 9	Nivel bajo circuito cerrado por medio de los contactos.	185
Cap.4: Fig. 10	Nivel alto, circuito abierto.	185
Cap.4: Fig. 11	Montaje de sensores de contacto.	186
Cap.4: Fig. 12	Hilo conductor en guante.	186
Cap.4: Fig. 13	Terminales de los sensores de contacto.	187
Cap.4: Fig. 14	Terminales de conexión para los sensores de contacto.	187
Cap.4: Fig. 15	Terminales de conexión en el guante.	188
Cap.4: Fig. 16	Montaje motor vibrador.	189
Cap.4: Fig. 17	Construcción de base para batería y PCB.	189
Cap.4: Fig. 18	Distribución de Pines Arduino Nano V3.	190
Cap.4: Fig. 19	Conexión serie de dos celdas LIPO.	190
Cap.4: Fig. 20	Cargando las baterías.	191
Cap.4: Fig. 21	PCB lado superior y lado inferior	191
Cap.4: Fig. 22	Elementos para ser montados en PCB.	192
Cap.4: Fig. 23	Montaje de módulos en PCB.	192
Cap.4: Fig. 24	Montaje de PCB y baterías sobre cajetín.	193
Cap.4: Fig. 25	Vista frontal “Guante Electrónico de Datos”.	195
Cap.4: Fig. 26	PCB en “Guante Electrónico de Datos” cara dorsal.	195
Cap.4: Fig. 27	Cara palmar “Guante Electrónico de Datos”.	196
Cap.4: Fig. 28	Conexión sensores de contacto a PCB “Guante Electrónico de Datos”.	196
Cap.4: Fig. 29	Módulos Xbee.	197
Cap.4: Fig. 30	Comunicación módulos Xbee.	198
Cap.4: Fig. 31	Posicionamiento del guante antes del encendido.	198
Cap.4: Fig. 32	Posicionamiento del guante antes del encendido.	199
Cap.4: Fig. 33	Icono de ejecutable.	199
Cap.4: Fig. 34	Evento Start.	200
Cap.4: Fig. 35	Evento Stop.	200
Cap.4: Fig. 36	Movimientos para el desplazamiento horizontal del cursor.	201
Cap.4: Fig. 37	Movimientos para el desplazamiento vertical del cursor.	202

Cap.4: Fig. 38	Transmisión de datos representados por cadenas de caracteres.	202
Cap.4: Fig. 39	Prueba de operación con imagen del esqueleto humano en 3D.	208
Cap.4: Fig. 40	Prueba de operación con imagen de un pie humano en 3D.	208

ÍNDICE DE TABLAS

Cap.1: Tabla. 1	Comparación entre Open source Software y Open source Hardware.	10
Cap.1: Tabla. 2	Cronología de la Realidad Virtual.	12
Cap.1: Tabla. 3	Movimientos del hombro.	23
Cap.1: Tabla. 4	Movimiento del codo.	24
Cap.1: Tabla. 5	Movimientos de la muñeca.	26
Cap.2: Tabla. 1	Tabla de Multiplicación de Cayley para cuaterniones.	49
Cap.2: Tabla. 2	Comparación de rendimiento por operaciones.	55
Cap.2: Tabla. 3	Principales características y aplicaciones de los acelerómetros considerando el margen de medida g.	65
Cap.2: Tabla. 4	Tabla comparativa de tipos de giroscopio.	75
Cap.2: Tabla. 5	Modelo ISO/OSI y Modelo estándar IEEE 802.	101
Cap.2: Tabla. 6	Comparativa de tecnologías Wireless.	110
Cap.3: Tabla. 1	Designación de pines Arduino Nano V3.	120
Cap.3: Tabla. 2	Designacion de pines Módulo XBee.	125
Cap.3: Tabla. 3	Función de pines del ADXL345.	129
Cap.3: Tabla. 4	Especificación de pines ITG-3200.	131
Cap.3: Tabla. 5	Designación de pines MPU-6050.	134
Cap.3: Tabla. 6	Interrupciones MPU-6050.	139
Cap.3: Tabla. 7	Asociación de puertos serial a ficheros de dispositivo.	144
Cap.3: Tabla. 8	Miembros de la estructura termios.	146
Cap.3: Tabla. 9	Eventos Xlib.	151
Cap.4: Tabla 1	Formato de visualización de datos para cada evento.	203
Cap.4: Tabla 2	Datos evento 1.	203
Cap.4: Tabla 3	Datos evento 2.	204
Cap.4: Tabla 4	Datos evento 3.	204
Cap.4: Tabla 5	Datos evento 4.	205
Cap.4: Tabla 6	Datos evento 5.	206
Cap.4: Tabla 7	Datos evento 6.	206
Cap.4: Tabla 8	Datos evento 7.	207
Cap.4: Tabla 9	Datos evento 8.	207

CAPÍTULO 1

1. INTRODUCCIÓN

1.1 PROYECTO

Desarrollo de un “Guante Electrónico de Datos” con sensores inerciales, herramientas Open Source y comunicación inalámbrica que interactúe con imágenes del cuerpo humano en 3D para el proyecto “Sistema de Entrenamiento virtual para medicina”.

1.1.1 PLANTEAMIENTO DEL PROBLEMA

En el país la educación superior en el área médica enfrenta limitaciones en el uso de los laboratorios para la manipulación de cadáveres, en los cuales no todos los estudiantes tienen la oportunidad de utilizar equipos y materiales por el gran número de alumnos, dificultándose la calidad del aprendizaje en el reconocimiento de las estructuras del cuerpo humano; habilidad fundamental y necesaria para la profesión médica. Por lo que se requiere la aplicación de modelos virtuales en los laboratorios de anatomía que proporcionen confianza y pericia, ahorrando tiempo, dinero, recursos y evitando riesgos de contaminación innecesarios al practicante.

En la actualidad el aprendizaje virtual permite al educando profundizar en la obtención de conocimiento en diferentes áreas de estudio como: ingeniería, aeronavegación, diseño, idiomas, medicina, psicología; por medio de programas multimedia, tutoriales, simuladores de una forma interactiva con el usuario; con gastos mínimos de infraestructura, tiempo, materiales, equipos, instructores, insumos, entre otros.

Un “Sistema de Entrenamiento Virtual para Medicina” permitirá al aprendiz del área de la salud familiarizarse en la manipulación de las partes del cuerpo humano, órganos y sistemas en forma virtual con imágenes en 3D a través de un “Guante Electrónico de Datos”, desarrollado al menor costo posible y utilizando Open Source (Hardware y Software).

Actualmente existen en el mercado internacional guantes electrónicos para la manipulación de imágenes en 3D con costos muy elevados, siendo este el principal inconveniente para su adquisición.

1.1.2 OBJETIVOS

1.1.2.1 GENERAL

Desarrollar un “Guante Electrónico de Datos” con sensores inerciales, herramientas Open Source y comunicación inalámbrica para la manipulación de imágenes del cuerpo humano en 3D, para el proyecto “Sistema de Entrenamiento Virtual para Medicina”.

1.1.2.2 ESPECÍFICOS

- Caracterizar la tecnología del guante electrónico existente en el mercado.
- Determinar el funcionamiento de los sensores (sensores de contacto, giroscopio y acelerómetro), dispositivos de procesamiento y el módulo inalámbrico para el desarrollo del “Guante Electrónico de Datos”.
- Seleccionar el hardware necesario a utilizarse en el desarrollo del “Guante Electrónico de Datos” con herramientas Open Source y comunicación inalámbrica.
- Evaluar diferentes tipos de materiales a usarse en el desarrollo del “Guante Electrónico de Datos.
- Mediante el armado de circuitos de prueba en un protoboard determinar de manera certera que elementos son óptimos a usarse en la aplicación final.

- Generar el diagrama esquemático para obtener el respectivo PCB (Print Circuit Board)¹ .
- Determinar el funcionamiento de los módulos de comunicación necesarios para la interrelación entre los sensores con la interfaz de usuario.
- Programar el módulo para la comunicación inalámbrica del guante electrónico de datos con la interfaz de usuario.
- Programar el dispositivo microcontrolador para el procesamiento de señales provenientes de los diferentes sensores para la manipulación física del guante.
- Ubicar la tarjeta de procesamiento de datos (PCB) y los sensores en un guante que sea: cómodo, ergonómico, liviano.
- Realizar prueba piloto del funcionamiento del “Guante Electrónico de Datos”.
- Elaborar placa electrónica final con elementos electrónicos de montaje superficial.

1.1.3 JUSTIFICACIÓN DEL PROYECTO

En la actualidad la tecnología informática a nivel mundial se ha convertido en una herramienta necesaria en todas las áreas de desarrollo. Cada vez existen mayores exigencias de aprendizaje y actualización de conocimientos de los estudiantes y profesionales en todas las áreas de estudio. Puntualmente, los sistemas computacionales hoy en día son una alternativa muy eficiente y económica para adquirir conocimiento o adiestramiento, por ejemplo: inglés, diseño, matemáticas, electricidad, etc.

La medicina usa también este sistema de aprendizaje para el adiestramiento de sus estudiantes, que por el limitado espacio y recursos de laboratorio no logran que todos adquieran la misma habilidad en la manipulación de las partes del cuerpo

¹ Medio para sostener mecánicamente y conectar eléctricamente componentes electrónicos, a través de rutas o pistas de material conductor, grabados en hojas de cobre laminadas sobre un sustrato no conductor, comúnmente baquelita o fibra de vidrio.

humano. Por tal motivo la creación de un “Guante Electrónico de Datos” será un apoyo en la etapa de estudio de esta disciplina que a su vez es requerido para el proyecto “Sistema de Entrenamiento virtual para medicina”.

Hoy en día en el mercado existen guantes electrónicos de datos para manipulación de imágenes en 3D que se comunican por medio del puerto serial y USB ocasionando dificultad e incomodidad de manipulación, por el peso y longitud de los cables de conexión, otra alternativa existente es la comunicación inalámbrica, pero su principal inconveniente es su alto costo en el mercado. Por tal motivo se hace necesario crear un “Guante Electrónico de Datos” con comunicación inalámbrica al menor costo posible.

El sistema planteado también busca la utilización de nuevas tecnologías basadas en sensores inerciales y comunicación inalámbrica. Este proyecto de investigación; constituye la base para el desarrollo de futuros proyectos relacionados con la navegación inercial y tecnología MEMS a través del uso de herramientas Open Source (Hardware y Software).

1.1.4 ALCANCE DEL PROYECTO

- El “Guante Electrónico de Datos” trabajará en la plataforma Linux
- Se desarrollará un “Guante Electrónico de Datos” para la mano derecha del usuario.
- El “Guante Electrónico de Datos” se comunicará por un medio inalámbrico con una aplicación gráfica de computadora que permite la manipulación de imágenes del cuerpo humano en 3D, desarrollada por otro grupo de estudiantes que son parte del proyecto de “Sistema de Entrenamiento Virtual para Medicina”. La comunicación se realizará mediante el intercambio de mensajes a través de cadenas de caracteres.
- Se utilizará sensores que reúnan las mejores características: sensores inerciales con tecnología MEMS y sensor de contacto.

- El “Guante Electrónico de Datos” manipulará las imágenes en 3D del proyecto de “Sistema de Entrenamiento Virtual para Medicina”, emulando un mouse aéreo para computador.
- La óptima respuesta que se conseguirá de la comunicación del “Guante Electrónico de Datos” con la aplicación que contiene las imágenes, dependerá del tamaño de la mano del usuario para obtener datos reales cuando se accionen los contactos ubicados en cada dedo.
- Dado los requerimientos que exige la aplicación se seleccionará un guante cuyas características de sus materiales brinden: flexibilidad, durabilidad, confort, y facilidad para el montaje de los diferentes sensores y su comunicación.
- Con el fin de permitir que el dispositivo sea utilizado de forma cómoda por la mayor cantidad de usuarios, se ha elegido un tamaño de guante promedio correspondiente a la talla 8.

1.1.5 METODOLOGÍA

La metodología es un recurso concreto que organiza el proceso de investigación, controlando cada uno de sus resultados, es decir; creando, acumulando y presentando posibles soluciones a los problemas que se puedan presentar dentro del marco de la investigación científica. El tipo de método de investigación que se escoge determina los pasos a seguir para el estudio, sus técnicas y métodos. Esto depende del tipo de investigación que se realizará. Antes de determinar el tipo de método² de investigación aplicado al proyecto, se señalan varios tipos de investigación que existen, y a partir de esto se identifica y describe el que se usará.

² Método es el conjunto de técnicas que sirven como la base teórica y la Metodología es la puesta en práctica de esas técnicas.

1.1.5.1 TIPOS DE INVESTIGACIÓN

▪ INVESTIGACIÓN BIBLIOGRÁFICA

Se fundamenta en la obtención y análisis de datos, apoyándose en fuentes de carácter documental que permite una visión panorámica de un problema.

▪ INVESTIGACIÓN DE CAMPO

Cuando los datos recogidos provienen directamente del hecho que se investiga, por medio de entrevistas, cuestionarios, encuestas y observaciones.

▪ INVESTIGACIÓN EXPERIMENTAL

En este tipo de investigación se somete el problema directamente a experimentación de manera de obtener de allí los datos.

En forma general, para que la metodología resulte eficiente debe ser disciplinada y sistemática, permitiendo analizar un problema en su totalidad. El método escogido para el presente trabajo es el denominado “Método analítico”, el cual satisface también los requerimientos del tipo de “Investigación Experimental”.

A continuación se describe el tipo de método usado en el proyecto.

1.1.5.2 MÉTODO ANALÍTICO

El método analítico es aquel método de investigación, que consiste en la desmembración de un todo, descomponiéndolo en sus partes o elementos para observar las causas, la naturaleza y los efectos. El análisis es la observación y examen de un hecho en particular. Es necesario conocer la naturaleza del fenómeno y objeto que se estudia para comprender su esencia. Este método nos permite conocer más del objeto de estudio, con lo cual se puede: explicar, hacer analogías, comprender mejor su comportamiento y establecer nuevas teorías.³

En el análisis se desintegra y descompone un todo en sus partes para estudiar en forma intensiva cada uno de sus elementos, su importancia se basa en comprender la naturaleza de cada elemento que conforman este todo y sus relaciones **(1)**.

³ Pág. 64. Ortiz Frida, García Maria del Pilar. Metodología de la Investigación. Editorial Limusa. México 2005

1.1.5.3 DESARROLLO TECNOLÓGICO

Un proyecto de desarrollo tecnológico soluciona metodológica y racionalmente un problema del mundo material satisfaciendo una necesidad o demanda específica de la sociedad.

A este proyecto se lo considerará como desarrollo tecnológico, porque está orientado a la obtención de productos tangibles, que resuelve problemas concretos y reales vinculados con el mundo artificial.

La metodología aplicada a este tipo de proyecto, permitirá abordar y resolver problemas prácticos y cubrir necesidades específicas durante la elaboración del proyecto “Guante Electrónico de Datos” con sensores inerciales, herramientas Open Source y comunicación inalámbrica que interactúe con imágenes del cuerpo humano en 3D para el proyecto “Sistema de Entrenamiento Virtual para Medicina”.

1.1.5.4 FASES DEL PROYECTO DE DESARROLLO TECNOLÓGICO

A continuación se presenta las fases usadas en la elaboración del “Guante Electrónico de Datos”.

▪ PLANTEAMIENTO DEL PROBLEMA

El planteamiento del problema es el punto de partida para construir un elemento que satisfaga una necesidad, presentando de una manera clara y directa la relación entre dos o más variables contenidas en el problema, permitiendo probar y encontrar las vías de solución o respuesta. Esta fase fue abordada y detallada en la página 1.

▪ BÚSQUEDA DE INFORMACIÓN

Una vez planteado el problema, en la siguiente fase se investiga y recolecta información de soluciones que se hayan dado a problemas similares y se organizan estos datos de forma sistemática, para entender su desarrollo en el tiempo.

▪ DISEÑO

Expresar por medio de dibujos y textos una propuesta para el proyecto, confeccionar un prototipo, calcular el costo de un producto.

▪ ORGANIZACIÓN Y GESTIÓN

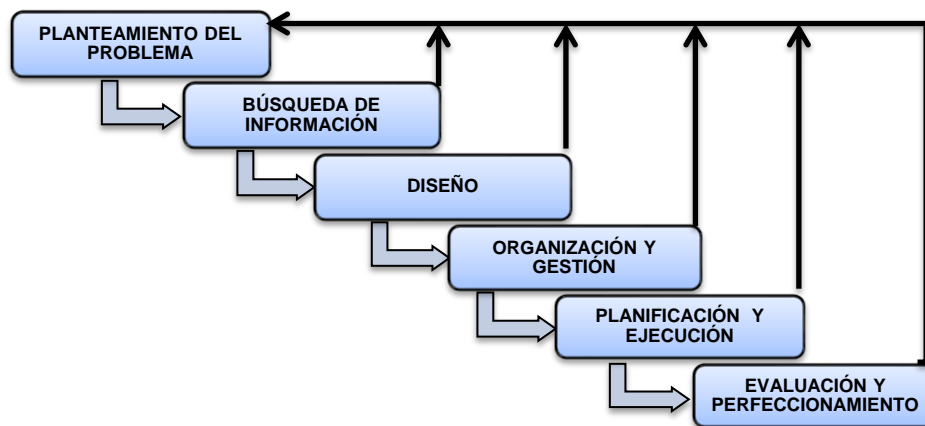
Asumir una función en el grupo a cargo de la realización del proyecto. Analizar varias alternativas antes de tomar decisiones. La buena organización hará ahorrar tiempo y dinero. En un documento deben reflejarse las fases de construcción, por orden cronológico, cómo deben ejecutarse, los materiales y herramientas a emplear, recursos disponibles (tanto materiales como humanos), el tiempo de construcción y su valor.

▪ PLANIFICACIÓN Y EJECUCIÓN

Se seleccionarán los materiales y equipos necesarios, proveedores y tiempo de entrega. Se interpretarán planos, documentación y especificaciones técnicas. Se proyectará el uso eficiente de los materiales y herramientas.

▪ EVALUACIÓN Y PERFECCIONAMIENTO

En donde se comparará el resultado obtenido con los objetivos iniciales. Se sugerirán cambios y mejoras en cada una de las fases anteriores.



Cap.1: Fig. 1 Fases del proyecto de desarrollo tecnológico.

FUENTE: AUTOR.

1.2 OPEN SOURCE

1.2.1 OPEN SOURCE SOFTWARE (FOSS)

Open Source Software hace referencia a las facultades que tiene el usuario de este tipo de software para usarlas, estudiarlas, modificarlas, adaptarlas, rediseñarlas y redistribuirlas; con la única condición de que luego de realizar cualquiera de las anteriores conductas, no introduzca ninguna restricción al producto así obtenido **(2)**.

1.2.2 OPEN SOURCE HARDWARE (OSHW)

Esencialmente, este es un problema de libertad. Nosotros los usuarios queremos tener la libertad de utilizar los objetos que poseemos para cualquier propósito y en combinación con cualquier otro objeto de software que elegir o crear. No debe limitarse la utilización sólo de la manera que el fabricante o alguna otra entidad externa estimen conveniente⁴.

Open Source Hardware es un término para dispositivos tangibles cuyas especificaciones y diagramas están disponibles al público de tal manera que se pueda modificar, distribuir y usar⁵.

Los términos de distribución de hardware de código abierto deben cumplir con los siguientes requisitos:

- Publicar la documentación incluyendo los archivos de los diseños, mismos que deben permitir su modificación y distribución.
- Especificar que porción del diseño es abierta en caso de que no se liberen todos los componentes.
- Ofrecer el software necesario para leer el archivo del diseño o la documentación suficiente de las funcionalidades requeridas, para que se pueda escribir el código open source del mismo fácilmente.

⁴ Lourens Veen (Open Hardware Foundation), 02-11-2011, http://wn.com/Open_Hardware_Foundation-Lourens_Veen.

⁵ Claudio Segovia (Vitacora de Claudio Segovia), 02-11-2011, <http://claudiosegovia.wordpress.com/tag/tecnologia/>.

- La licencia no debe restringir a que se venda o comparta la documentación necesaria.
- Ofrecer una licencia que permita producir derivados y modificaciones, además su re-distribución bajo la licencia original, así como su venta y manufactura.
- La licencia no debe restringir a ningún campo o actividad.

	FUENTE	PROCESO DE PRODUCCIÓN	PRODUCTO
SOFTWARE	CÓDIGO (LÓGICO)	COMPILACIÓN	EJECUTABLE (LÓGICO)
HARDWARE	DISEÑO (LÓGICO)	FABRICACIÓN	PRODUCTO (FÍSICO) \$\$

Cap.1: Tabla. 1 Comparación entre Open source Software y Open source Hardware.

FUENTE: (3).

1.2.2.1 ARDUINO

Es una plataforma Open Source que está fundamentada en hardware y software, basada en un microcontrolador y un entorno de desarrollo, que se utiliza tanto para el aprendizaje como también para automatizar cualquier proceso en el mundo real. Sus diseños son libres, los esquemas y documentación están publicados en el sitio Web **(4)**.

El hardware consiste en una placa con microprocesador Atmel AVR, el software es un entorno de desarrollo basado en el lenguaje de programación Processing/Wiring, las placas Arduino están abiertas al público; esta fue la razón para escoger este tipo de tecnología. Constituyéndose en la base que permite una interacción integral, tanto de la parte de comunicación como del sensado de las posiciones en el espacio tridimensional del “Guante Electrónico de Datos”.

1.3 REALIDAD VIRTUAL

El constante avance tecnológico que en la actualidad, permite manejar grandes cantidades de información, ha generado conocimiento en la misma proporción en las diferentes áreas de desarrollo en la sociedad. Tal es el caso de los sistemas

multimediales; que ofrecen la combinación de texto, audio y video en un sólo documento brindando un producto atractivo y eficiente para los usuarios.

La tecnología de realidad virtual se presenta actualmente para añadir a los sistemas multimediales: percepciones visuales, auditivas, táctiles, gustativas y olfativas permitiendo una interactividad más eficiente entre la máquina y el usuario. Esta tecnología prueba ser eficiente en la medicina, educación, arquitectura, deporte, recreación, etc.

1.3.1 DEFINICIÓN

La realidad virtual es un medio que genera un ambiente artificial interactivo entre la computadora y el usuario, proporcionando al individuo la sensación y percepción de un entorno natural en tiempo real.

Según, Josep Gurri⁶: “La Realidad Virtual es una representación vectorial que tiene como resultado una imagen, generada por ordenador. Esta imagen puede ser estática, en movimiento, o interactiva, según el formato y el soporte del proyecto”.

1.3.2 HISTORIA

El primer antecedente histórico de la realidad virtual fue en 1963, Iván Sutherland, elaboró el software Sketchpad para su tesis de doctorado en el MIT(Massachusetts Institute of Technologic) **(5)**.

⁶ Josep Gurri en una entrevista a la revista MOSAIC. <http://mosaic.uoc.edu/2006/01/23/josep-gurri/>

CRONOLOGÍA DE LA REALIDAD VIRTUAL

1965 Surge el concepto de Realidad Virtual por Iván Sutherland, en un artículo titulado "The Ultimate Display", describiendo el concepto básico de la Realidad Virtual.
1966 Sutherland creó el primer casco visor de Realidad Virtual.
1968 Iván Sutherland y David Evans crean el primer generador de escenarios con imágenes tridimensionales, datos almacenados y aceleradores. En este año se funda también la sociedad Evans & Sutherland.
1971 Redifon Ltd en el Reino Unido comienza a fabricar simuladores de vuelo con displays gráficos.
1972 General Electric, desarrolla el primer simulador computarizado de vuelo.
1977 Dan Sandin y Richard Sayre inventan un guante sensitivo a la flexión.
1980 Jaron Lanier acuñó la expresión "Realidad Artificial".G. J. Grimes, asignado a Bell Telephone Laboratories, patentó un guante para introducir datos.
1982 Thomas Zimmerman patentó un guante para introducir datos basados en sensores ópticos.
1984 Mc Greevy y Humphries desarrollaron el sistema VIVED (Virtual Visual Environment Display) para los futuros astronautas en la NASA.
1986 Existen ya laboratorios como el de la NASA, Universidad de Tokio, Boeing, Sun Microsystems, Intel, IBM y Fujitsu dedicados al desarrollo de la tecnología VR.
1987 La NASA utilizando algunos productos comerciales, perfecciona la primera realidad sintetizada por computadora mediante la combinación de imágenes estéreo, sonido 3-D, guantes, etc.
1989 Autodesk, Inc. Hizo una demostración de su PC basada en un sistema CAD de Realidad Virtual, Ciberespacio, en SIGGRAPH'89.
1990 Surge la primera compañía comercial de software VR, Sense8. Ofrece las primeras herramientas de software para VR.
1994 La Sociedad de Realidad Virtual fue fundada. IBM y Virtuality anunciaron el sistema V-Space.

Cap.1: Tabla. 2 Cronología de la Realidad Virtual.

FUENTE: (6).

1.3.3 CLASIFICACIÓN

▪ REALIDAD VIRTUAL DE ESCRITORIO

Aquellas aplicaciones que muestran imágenes 2D o 3D en un monitor de computadora, que usualmente necesitan de lentes de cristal y pantallas LCD. Ejemplos; juegos PC, PlayStation.

▪ REALIDAD VIRTUAL EN SEGUNDA PERSONA

El usuario es introducido en el mundo virtual como parte de la escena, cuyas respuestas se dan en tiempo real mediante el uso de guantes o cascos conectados al usuario. Este logra verse a sí mismo dentro de la escena, siendo esta la diferencia con sistemas de inmersión completa.

▪ TELEPRESENCIA

Sistemas equipados con cámaras, micrófonos y dispositivos táctiles que permiten al usuario experimentar una situación remota. Ejemplo: telecirugía, microcirugía, exploración espacial, etc.

▪ INMERSIÓN

Sumergen al usuario en un mundo virtual mediante el uso de cascos visuales y auditivos, rastreadores de posición y movimiento. Ejemplo: sistema de videojuegos, arquitectura virtual, etc. **(7)**.

1.3.4 CARACTERÍSTICAS PRINCIPALES DE LA REALIDAD VIRTUAL

A) INMERSIÓN

Propiedades de los aparatos que aíslan los sentidos del usuario (visual, auditivo y táctil) lo suficiente para que se sienta transportado a otro lugar creando una ilusión generada por computadora. La característica de la percepción es asociada con el grado de inmersión que el usuario siente.

Atributos de la inmersión

- La habilidad o capacidad para enfocar la atención del usuario.
- Capaz de convertir una base de datos almacenada en la computadora en experiencias.



Cap.1: Fig. 2 Simulador de realidad virtual inmersivo.

FUENTE: (8).

B) INTERACTIVIDAD

Permite al usuario el control del sistema creado, es decir que el sistema responda a los estímulos de la persona que lo utiliza creando una interdependencia entre ellos. La realidad virtual no es un sistema pasivo, es esto una diferencia muy importante, cuando se compara con el ver televisión o una película en la primera fila del cine, en la que el usuario no tiene una interacción con el sistema.

C) INTENSIDAD DE INFORMACIÓN

Es la noción que el mundo virtual ofrece al usuario como la telepresencia y entidades artificiales que poseen cierto grado de comportamiento inteligente.

Dentro de estas tres características se agrupan la manipulación, presencia remota teleoperación, percepción, etc. Todas estas son parte de la realidad virtual que se insertan en las “tres l’s” que Michael Heim propone: inmersión, interactividad e intensidad de información **(9)**.

1.3.5 DISPOSITIVOS

A) HARDWARE

Elementos externos que permiten la interacción con el mundo virtual y generan su propio entorno virtual.

Dispositivos de entrada

- **Elementos de control:** Guantes y trajes de datos, joysticks 3D, etc.
- **Rastreadores de posición y movimiento:** Controlan la visión del entorno virtual y su posición en el espacio un ejemplo es el giroscopio.

Dispositivos de salida

- **Generador de imágenes** Lentes de obturación, Cascos, visores, sistemas binoculares, lentes estereoscópicos.
- **Generador de Sonidos** Cascos auditivos, sonido tridimensional

B) SOFTWARE

Programas para el desarrollo de simulaciones de mundos virtuales fundamentados en los conceptos de computación gráfica divididos en varias clases y librerías de programas.

Herramientas de autoría en mundos virtuales

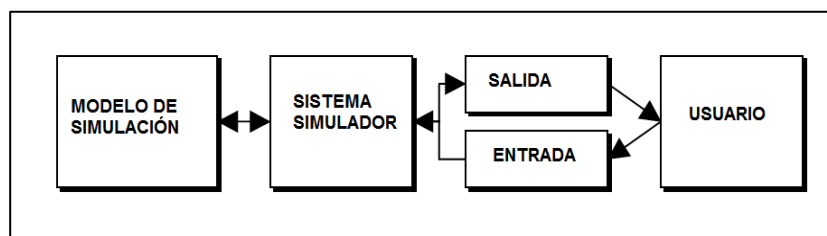
Son aplicaciones informáticas de carácter multimedia, que combinan documentos digitales, imágenes, sonido, videos permitiendo a sus usuarios crear sus propios proyectos. Ejemplo: SuperscapeVRT, VRCreator, Meme, COVISE, 3DMax, Blender.

API de realidad virtual

Funciones y herramientas que una determinada biblioteca pone a disposición para que sean utilizadas por otro software para crear gráficos en 2D y 3D. Como ejemplo se cita: Dive, Render Ware, 3DR, World Toolkit, VRML, Lenguajes de alto nivel.

1.3.6 SISTEMA DE REALIDAD VIRTUAL

El modelo genérico de un sistema de realidad virtual esta formado por las etapas más comunes que intervienen en la interacción del usuario con el sistema virtual, estas etapas son: el modelo de simulación, el cual está representado en la computadora y asociado a un sistema visual o auditivo, el sistema simulador que es el escenario donde se produce la interacción con el usuario mediante los dispositivos de entrada y salida.



Cap.1: Fig. 3 Modelo genérico de un Sistema de Realidad Virtual.

FUENTE: (10).

1.3.7 APLICACIONES

Medicina Facilita la manipulación de órganos internos e intervenciones quirúrgicas, también la creación de pacientes virtuales para poner en práctica las habilidades de los estudiantes, un ejemplo de esto es: “Sistema de Entrenamiento virtual para medicina”.

Ciencia Para el estudio de los sistemas de partículas y moléculas.

Diseño Herramientas de CAD para diseñar prototipos y probarlos en ambientes virtuales.

Defensa Simuladores de vuelo, entrenamiento militar.

Arquitectura Navegación por futuros edificios.

Enseñanza Facilita la atención de los estudiantes mediante su inmersión en mundos virtuales.

1.4 GUANTE ELECTRÓNICO DE DATOS

Un guante electrónico de datos es un dispositivo interactivo que está equipado con sensores electrónicos, cuya finalidad es la de servir como periférico de entrada, principalmente en entornos de realidad virtual para controlar movimientos en un entorno tridimensional, los guantes son equipados con sensores de tacto o sensibilidad. Cabe anotar que un guante de datos no permite “tocar” el mundo virtual, sino que provoca la sensación al usuario de que se encuentra en este entorno tridimensional (inmersión).

El empleo del guante permite al usuario una mejor interacción con el sistema ya que posee sensores que miden su posición y orientación.

El primer guante de datos fue desarrollado en los laboratorios Bell de AT&T por el Dr. G. Grimes, denominado “Digital Data Entry Glove”; equipado con sensores de flexión en los dedos, sensores táctiles en la parte de la yema de los dedos y con sensores de posicionamiento espacial.

1.4.1 VERSIONES ACTUALES

DG5 VHAND 2.0

Es un equipo completo, equipado con 5 sensores, miden el movimiento de los dedos, mientras que el acelerómetro detecta los movimientos y orientación de la mano. La transmisión se realiza mediante protocolo estándar RS232 con 115200 BPS. Equipada con una batería de 20mA, necesita un voltaje de entrada de 3.3V a 5V. Sus usos están dirigidos hacia: robótica, realidad virtual, juegos. El costo en el mercado es de \$ 992 (11).



Cap.1: Fig. 4 DG5 VHAND 2.0.

FUENTE: (11).

DATA GLOVE 5DT

Fabricado en licra y diseñado para la animación en tiempo real y captura de movimiento, alta tasa de transmisión de datos e igual calidad de los mismos. Un sensor en cada dedo para medir su flexión, se comunica con la PC vía USB. Dispone de conexión con el puerto RS232 independiente de la plataforma. Ofrece una resolución de 8bits en la flexión. Su costo en el mercado es de \$ 1425 **(11)**.



Cap.1: Fig. 5 DATA GLOVE 5DT.

FUENTE: (11).

5DT DATA GLOVE 5 MRI

La serie RMI de guante de datos 5DT esta destinado para ambientes de proyección de imagen de resonancia magnética (MRI). El guante se comunica por medio de una caja de control vía fibra óptica; esta caja se conecta con la PC mediante un cable que esta conectado al puerto serie RS232 (independiente de la plataforma). Existen modelos de 5 y 14 sensores, Tiene un costo de \$ 4177 **(11)**.



Cap.1: Fig. 6 5DT DATA GLOVE 5 MRI.

FUENTE: (11).

CYBER GLOVE II

Cuenta con 18 sensores: plegado, captura, medición, arqueado de la mano, flexión y captura de la muñeca. El sistema de captura de movimiento es utilizado en

aplicaciones reales como realidad virtual biomecánica y animación. Posee un software programable para el conmutador sobre la muñeca que agrega capacidad adicional de entrada y salida. Tiene un costo de \$12295 **(12)**.



Cap.1: Fig. 7 CYBER GLOVE II

FUENTE: (13).

CYBERGRASP EXOSKELETON

Este es un dispositivo tipo exoesqueleto, el cual se coloca sobre el CyberGlobe. Este sistema permite al usuario sentir el tamaño y la forma de los objetos 3D, es decir permite literalmente sentir el objeto manipulado por medio de una red de tendones conectados a las yemas de los dedos a través del exoesqueleto. Precio en el mercado \$ 80.4337 **(13)**.



Cap.1: Fig. 8 CYBERGRASP EXOSKELETON.

FUENTE: (13).

ACCELEGLOVE

Este es un guante de datos de código abierto, programable que registra los movimientos de la mano y los dedos. Software programable en Java. Anthro Tronix desarrolló inicialmente el guante junto al departamento de defensa de los EEUU, para funciones relacionadas con el control robótico. Bajo la punta de cada dedo existe un acelerómetro los cuales detectan la orientación tridimensional de los dedos y la palma con relación a la gravedad terrestre **(14)**.



Cap.1: Fig. 9 ACCELEGLOVE

FUENTE: (14).

1.5 ANATOMÍA DE LA MANO HUMANA

La mano es una de las herramientas más maravillosas y difíciles de imitar, pertenece al sentido del tacto que es un conjunto de mecanismos sensoriales que el ser humano usa para interactuar con su entorno, este sentido es capaz de palpar los objetos, modificar su estructura, construir, destruir, etc. La persona al tocar un objeto obtiene información sobre este, como: dureza, tamaño, volumen, peso, etc.

Desde el punto de vista fisiológico, la mano representa la extremidad efectora del miembro superior. Sin embargo, no es tan solo un órgano de ejecución, es también un receptor sensorial extremadamente sensible y preciso cuya información es indispensable para retroalimentar su propia acción.

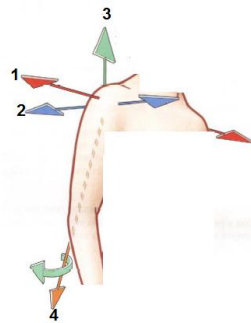
Para describir los movimientos finales de la mano es necesario considerar las articulaciones anteriores que ayudan a efectuar el movimiento final de esta, estas articulaciones son: el hombro, el codo y la muñeca.

1.5.1 EL HOMBRO

El hombro es un conjunto de articulaciones con mucha movilidad que une el torso con el brazo facilitándole a este, el desplazamiento en tres grados de libertad. El conjunto de articulaciones trabajan de forma coordinada permitiendo movimientos amplios y en muchas direcciones.

EJES PRINCIPALES DEL HOMBRO

1. Eje transversal usado en el movimiento de flexoextensión en el plano sagital⁷.
2. Eje anteroposterior usado en el movimiento de abducción-aducción en el plano transversal.
3. Eje vertical usado en los movimientos de flexión y extensión.
4. Eje longitudinal del húmero permite la rotación externa-interna del brazo.



Cap.1: Fig. 10 Principales ejes de movimiento del hombro.

FUENTE: (15).

1.5.1.1 MOVIMIENTO DEL HOMBRO

El hombro permite tres movimientos básicos: abducción-aducción, flexión-extensión y rotación interna y externa.

▪ ABDUCCIÓN Y ADUCCIÓN

Estos movimientos ocurren cuando el brazo se aleja del cuerpo (abducción) y cuando se acerca nuevamente (aducción).



Cap.1: Fig. 11 Movimiento de abducción y aducción del hombro.

FUENTE: (16).

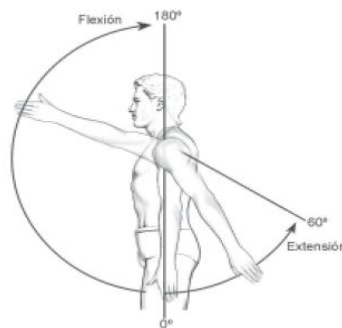
⁷ El plano sagital en anatomía es aquel plano perpendicular al suelo y paralelo al plano medio sagital, y que divide al cuerpo en mitades izquierda y derecha.

- **FLEXIÓN**

Movimiento alrededor del eje transversal cuando se eleva el brazo hacia delante de la posición anatómica.

- **EXTENSIÓN**

Movimiento opuesto a la flexión alrededor del eje transversal, desplaza el miembro superior hacia atrás de la posición anatómica.



Cap.1: Fig. 12 Movimiento de flexión y extensión del hombro.

FUENTE: (17).

1.5.1.2 ROTACIÓN DEL HOMBRO

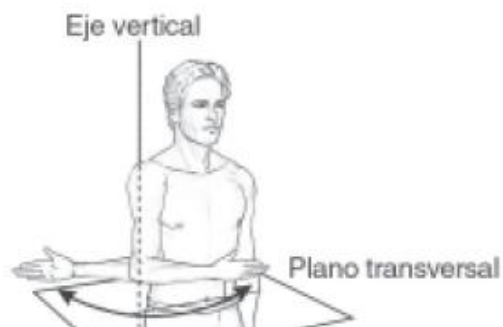
Se ejecuta a través del eje longitudinal del húmero, utiliza el tercer grado de libertad (3GDL). Se lo define como la rotación del miembro sobre su propio eje.

- **ROTACIÓN EXTERNA**

El movimiento en el plano transversal que desplaza una parte del cuerpo hacia fuera se denomina rotación externa.

- **ROTACIÓN INTERNA**

El movimiento en el plano transversal que desplaza una parte del cuerpo hacia dentro se denomina rotación interna.



Cap.1: Fig. 13 Movimiento de rotación del hombro derecho.

FUENTE: (17).

Movimientos	Eje	Plano	Amplitud
Flexo – Extensión	transversal	sagital	Flexión 0° - 180° Extensión 0° - 50°
Aducción – Abducción	anteroposterior	Coronal	Aducción 0° - 30° Abducción 0° - 180°
Rotación interna – Rotación externa	Longitudinal de húmero	Sagital	Rot. Interna 0° - 90° Rot. Externa 0° - 90°

Cap.1: Tabla. 3 Movimientos del hombro.

FUENTE: (18).

1.5.2 EL CODO

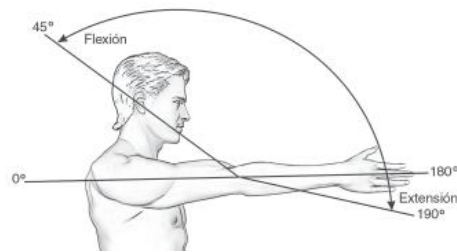
El codo es la articulación intermedia del miembro superior que permite la orientación en los tres planos del espacio con la ayuda del hombro y desplazando la mano del cuerpo.

1.5.2.1 MOVIMIENTOS DEL CODO

Anatómicamente el codo cuenta con una sola articulación sin embargo se define otros movimientos que se logra con la ayuda del hombro

▪ FLEXIÓN- EXTENSIÓN

Flexión es el movimiento que permite acercar las caras anteriores del brazo y del antebrazo, el retorno del movimiento de flexión se denomina extensión.

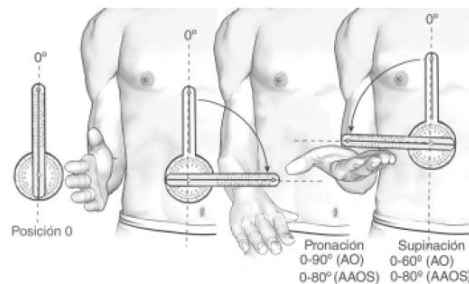


Cap.1: Fig. 14 Flexión y extensión del codo.

FUENTE (17).

▪ ROTACIÓN O PRONO-SUPINACIÓN

Es un movimiento que se produce a nivel del codo, de la membrana interósea y de la muñeca.



Cap.1: Fig. 15 Movimientos del codo.

FUENTE (17).

Movimientos	Eje	Plano	Amplitud
Flexo - Extensión	transversal	sagital	Flexión 0° - 150° Extensión 0° - 15°
Prono – Supinación	Longitudinal del antebrazo		Pronación 0° - 90° Supinación 0° - 90°

Cap.1: Tabla. 4 Movimiento del codo.

FUENTE: (18).

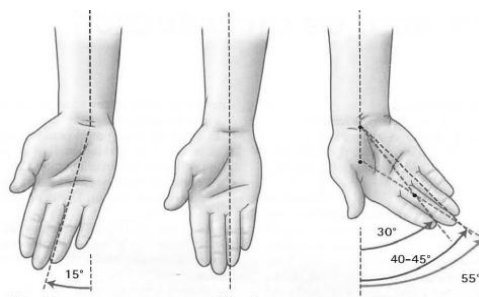
1.5.3 LA MUÑECA

La muñeca cumple una labor muy importante en la mano en lo que a movimientos y posición óptima para la prensión se refiere. La muñeca posee dos grados de libertad que otorgan a la mano los movimientos de abducción-aducción y flexión-

extensión. Combinados, ejecutan el movimiento de circunducción, siendo este el tercer GDL que recorre una superficie en forma de cono no regular y su vértice corresponde al centro de la muñeca **(15)**.

1.5.3.1 ABDUCCIÓN-ADUCCIÓN DE LA MUÑECA

Se mide a partir de la posición anatómica: el eje de la mano representada por el tercer metacarpo y el tercer dedo. La abducción o inclinación radial no sobrepasa los 15° . La aducción o inclinación cubital es de 45°

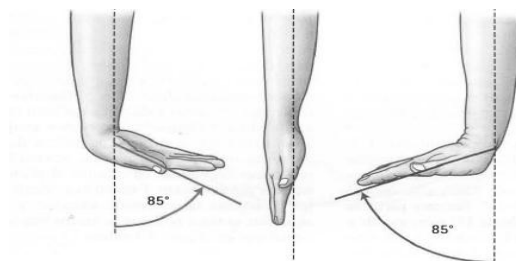


Cap.1: Fig. 16 Movimientos de abducción y aducción de la muñeca.

FUENTE: (15).

1.5.3.2 FLEXIÓN-EXTENSIÓN DE LA MUÑECA

Se mide a partir de la posición anatómica: muñeca alineada, cara dorsal de la mano en la prolongación de la cara posterior del antebrazo. La flexión activa es de 85° es decir apenas alcanza los 90° . La extensión también es de 85° y no alcanza los 90° .

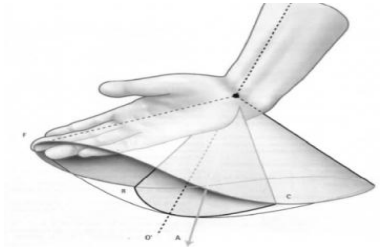


Cap.1: Fig. 17 Movimiento Flexión y Extensión de la muñeca.

FUENTE: (15).

1.5.3.3 MOVIMIENTO DE CIRCUNDICCIÓN DE LA MUÑECA

Este movimiento es la combinación de los movimientos de abducción-aducción y flexión-extensión. Cuando el movimiento de circundicción alcanza su máxima amplitud, el eje de la mano describe una superficie cónica en el espacio, denominado cono de circundicción.



Cap.1: Fig. 18 Movimiento de circundicción de la muñeca.

FUENTE: (15).

Movimientos	Eje	Plano	Amplitud
Flexo - Extensión	transversal	sagital	Flexión 0° - 85° Extensión 0° - 45°
Abducción Aducción	Anteroposterior	coronal	Pronación 0° - 45° Supinación 0° - 15°

Cap.1: Tabla. 5 Movimientos de la muñeca.

FUENTE: (18).

1.5.4 LA MANO

Está compuesta por huesos, músculos, tendones, ligamentos y un sistema nervioso sensorial, que funcionan de forma coordinada; obteniendo una gran cantidad de movimientos y acciones, siendo la disposición del pulgar con respecto a los otros dedos de la mano la que ayuda en las tareas de agarre, manipulación y corte.

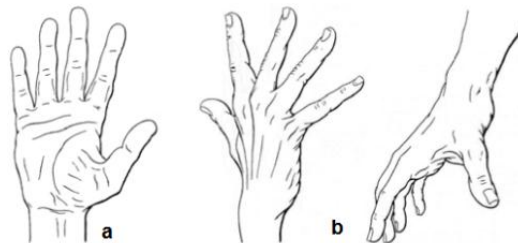
La mano para su movimiento también depende de la ayuda del brazo y la muñeca para su traslación y rotación.

La mano humana tiene un número alto de grados de libertad, alta relación fuerza-peso y un sistema sensorial complejo. Cada dedo posee dos grados de libertad en la

base con excepción del pulgar que tiene cinco grados de libertad y 2 articulaciones tipo bisagra que proporcionan los movimientos de flexión y extensión.

1.5.4.1 TOPOGRAFÍA BÁSICA

En la parte palmar, la mano esta formada por una palma, un pulgar y cuatro dedos. En la cara dorsal se compone de: red venosa, los dedos y los tendones externos.

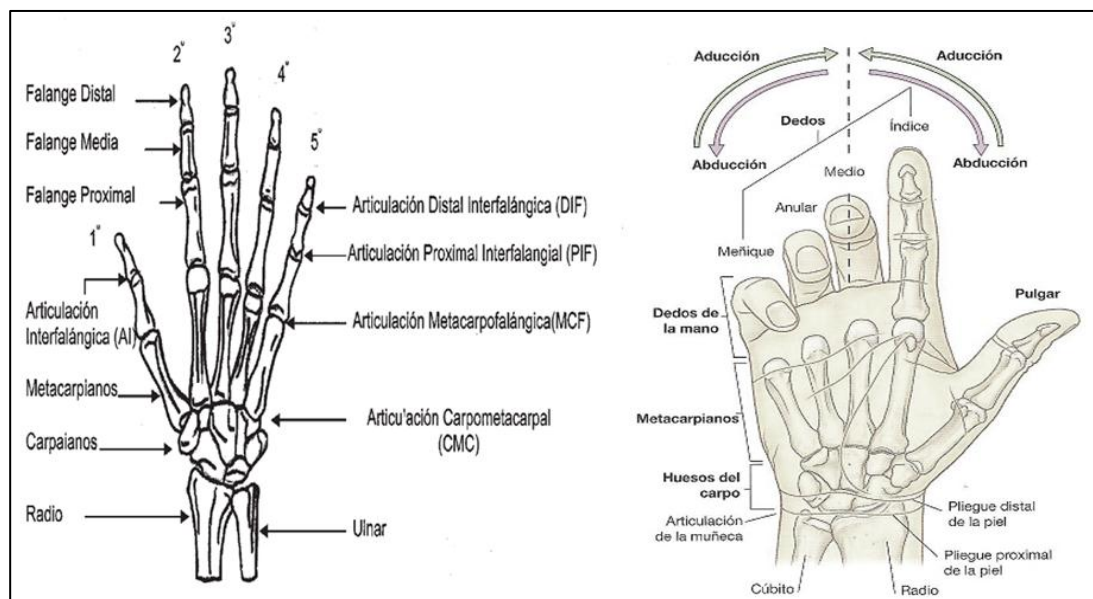


Cap.1: Fig. 19 a) Cara palmar y b) Cara dorsal.

FUENTE: (15).

1.5.4.2 TOPOGRAFÍA ÓSEA

La mano humana se divide en tres regiones secundarias: huesos del carpo, metacarpianos y dedos.



Cap.1: Fig. 20 Topografía de la mano.

FUENTE: (19).

▪ **CARPO**

Lo forman 8 huesos cortos dispuestos en dos filas superpuestas entre sí que constituyen los huesos de la muñeca que son: escafoide, semilunar, piramidal, pisiforme, trapecio, trapezoide, grande y ganchoso.

▪ **METACARPO**

Parte de la mano comprendida entre el carpo y los dedos, los huesos que lo conforman están relacionados cada uno con un dedo, se enumeran del 1 al 5 siendo el primer metacarpiano relacionado con el dedo pulgar.

▪ **METACARPOFALÁNGICAS**

Son cada una de las prolongaciones que se encuentran en la mano metacarpo-falanges, es decir, son las falanges, que constituyen los huesos de los dedos. El pulgar solo tiene dos falanges, mientras que el resto de los dedos tiene tres.

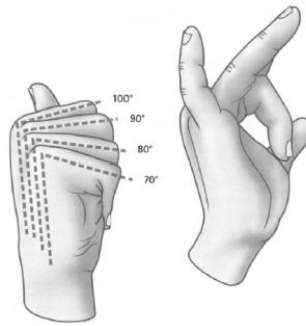
Los huesos del carpo y los metacarpianos de los dedos índice, medio, anular y meñique tienden a actuar como una unidad y constituyen la mayor parte del esqueleto óseo de la palma.

El metacarpiano del pulgar funciona de forma independiente y tiene más flexibilidad en la articulación carpometacarpiana para conseguir la oposición del pulgar a los otros dedos.

1.5.4.3 MOVIMIENTOS METACARPOFALÁNGICAS

▪ **FLEXIÓN**

Es aproximadamente de 90° desde el dedo índice, aumentando progresivamente hasta el quinto dedo. También cabe señalar que la flexión aislada de un dedo esta limitada por la tensión del ligamento palmar interdigital.

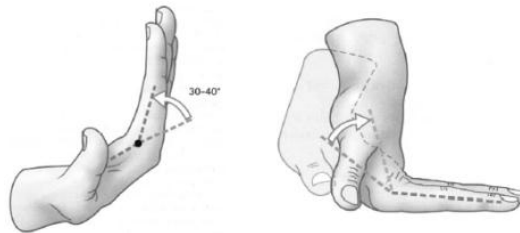


Cap.1: Fig. 21 Movimiento de Metacarpofalange.

FUENTE: (15).

▪ EXTENSIÓN

Varía según el individuo, alcanza de 30 a 40° y hasta casi 90° con una gran laxitud ligamentosa.



Cap.1: Fig. 22 Extensión de la mano.

FUENTE: (15).

De todos los dedos, excepto el dedo pulgar, el dedo índice es el que posee la mayor amplitud de movimiento en sentido lateral 30°; denominando a estos movimientos abducción cuando se separa del origen y aducción cuando regresa al origen.



Cap.1: Fig. 23 Abducción del dedo índice.

FUENTE: (15).

La circunducción, como ya se mencionó es la combinación de los movimientos de abducción-aducción y extensión-flexión en distintos grados. Estos movimientos quedan circunscritos al interior del cono de circunducción.



Cap.1: Fig. 24 Circunducción del dedo índice.

FUENTE: (15).

1.5.5 EL PULGAR

La oposición es el principal movimiento del pulgar; es la facultad de desplazar la yema del pulgar en contacto con la yema de uno de los otros cuatro dedos para construir una pinza pulgodigital. De esta manera conoce una gama de oposiciones que efectúan una gran variedad de presas y de acciones según el número de dedos implicados (15).

Toda la funcionalidad del pulgar implica la relación con los otros dedos y viceversa.

1.5.5.1 COMPONENTES DE MOVIMIENTO DEL PULGAR

▪ ANTEPOSICIÓN

Movimiento que desplaza el pulgar por delante del plano de la palma de la mano, involucra la articulación trapeciometocarpiana (TMC).



Cap.1: Fig. 25 Anteposición del pulgar.

FUENTE: (15).

▪ FLEXIÓN

Desplaza toda la columna del pulgar hacia dentro denominado también abducción, esta flexión involucra a todas las articulaciones del pulgar.

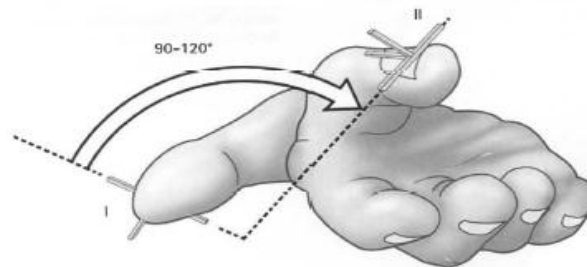


Cap.1: Fig. 26 Flexión del pulgar.

FUENTE: (15).

▪ PRONACIÓN

Componente esencial de la oposición del pulgar, cambio de movimiento de la última falange del pulgar que mira en direcciones diferentes, según su grado de rotación, permitiendo a las yemas de los dedos contactarse con el pulgar.



Cap.1: Fig. 27 Pronación del pulgar.

FUENTE: (15).

▪ OPOSICIÓN

La oposición asocia los tres componentes ya descritos: anteposición, flexión, pronación. Las articulaciones metacarpofalángica e interfalángica permiten distribuir la oposición sobre cada uno de los últimos cuatro dedos tomando; como eje el dedo pulgar para seleccionar el dedo al que se va a oponer, siendo las relaciones de oposición con el pulgar: pulgar-índice, pulgar-meñique, pulgar-corazón, pulgar-anular.

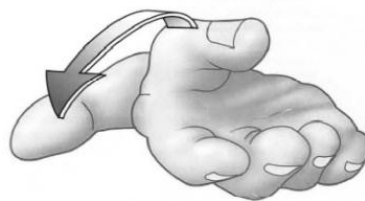


Cap.1: Fig. 28 Oposición pulgar-meñique.

FUENTE: (15).

▪ CONTRAPOSICIÓN

La contraposición proporciona a la mano la acción de soltar o prepararse para tomar objetos más voluminosos.

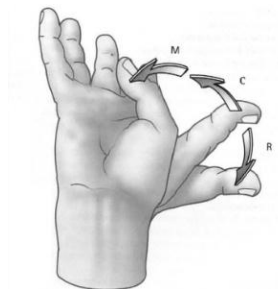


Cap.1: Fig. 29 Movimiento de contraposición.

FUENTE: (15).

Los nervios motores que intervienen en las acciones de oposición y contraposición son los siguientes:

- El nervio radial R para la contraposición.
- El nervio cubital C para el cierre de las presas.
- El nervio mediano M en el caso de la oposición.

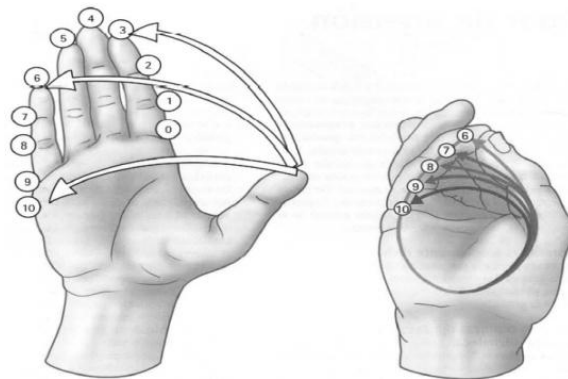


Cap.1: Fig. 30 Nervios motores que accionan la oposición y contraposición.

FUENTE: (15).

▪ TOTAL OPOSICIÓN

Pruebas de oposición y contraposición donde la mano del sujeto sirve de sistema de referencia: el pulgar, parte de la máxima separación y recorre el trayecto mayor de oposición en contacto sucesivo con la yema de los restantes dedos.

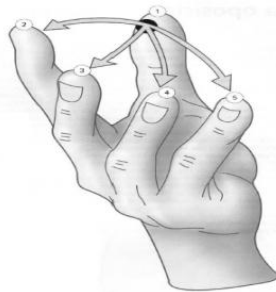


Cap.1: Fig. 31 Total oposición del pulgar.

FUENTE: (15).

1.5.6 TOPOGRAFÍA FUNCIONAL

La funcionalidad de la mano esta representada en un 40% por la presencia del dedo pulgar, este permite realizar pinzas con cada uno de los dedos. El pulgar juntamente con el índice y el medio se encargan de hacer la toma de precisión, mientras el pulgar con el dedo anular y meñique se usan para la toma palmar (asegurando la firmeza del puño).



Cap.1: Fig. 32 Funcionamiento del pulgar con todos los dedos de la mano.

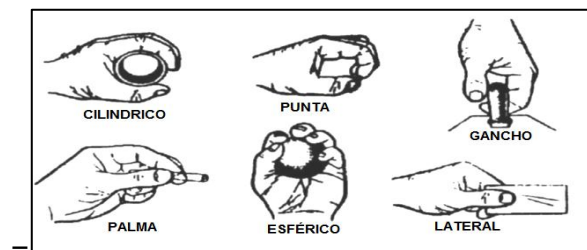
FUENTE: (15).

1.5.7 FUNCIÓN PRENSIL

La función prensil de la mano es la que permite agarrar un objeto y también sostenerlo.

1.5.7.1 AGARRE DE FUERZA

La acción de tomar con la palma de la mano y los dedos, proporciona fuerza y resta precisión. Las técnicas de agarre fueron clasificadas por Schlesinger en seis categorías: agarre cilíndrico, de punta, de gancho, de palma, esférico y de lado.



Cap.1: Fig. 33 Tipos de agarre de la mano.

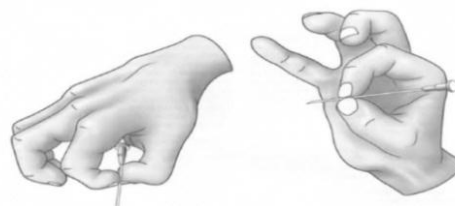
FUENTE: (20).

1.5.7.2 AGARRE DE PRECISIÓN DIGITAL

En este tipo de agarre se manipulan objetos de menor tamaño con la ayuda de los dedos y el pulgar. El pulgar se opone a la palma de la mano en un rango amplio de movimientos.

1.5.7.2.1 OPOSICIÓN TERMINAL O TERMINOPULPEJO

Es la mas fina y precisa, permite sujetar un objeto de pequeño calibre o coger un objeto muy fino con el menor esfuerzo y alteración de la mano.

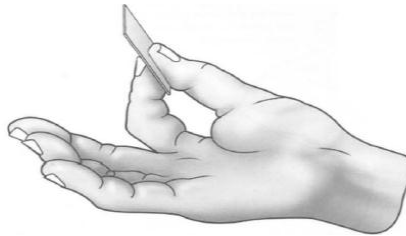


Cap.1: Fig. 34 Oposición terminal.

FUENTE: (15).

1.5.7.2.2 OPOSICIÓN SUBTERMINAL

Permite sujetar objetos relativamente gruesos: un lápiz, una hoja de papel, en este tipo de agarre el pulgar y el otro dedo se oponen por la cara palmar de la yema o pulpejo.



Cap.1: Fig. 35 Oposición subterminal.

FUENTE: (15).

1.5.7.2.3 OPOSICIÓN SUBTERMINOLATERAL O PULPOLATERAL

Como cuando se sujeta una moneda el agarre es menos fino aunque sigue siendo solido. La cara palmar de la yema del pulgar contacta con la cara externa de la primera falange del dedo índice

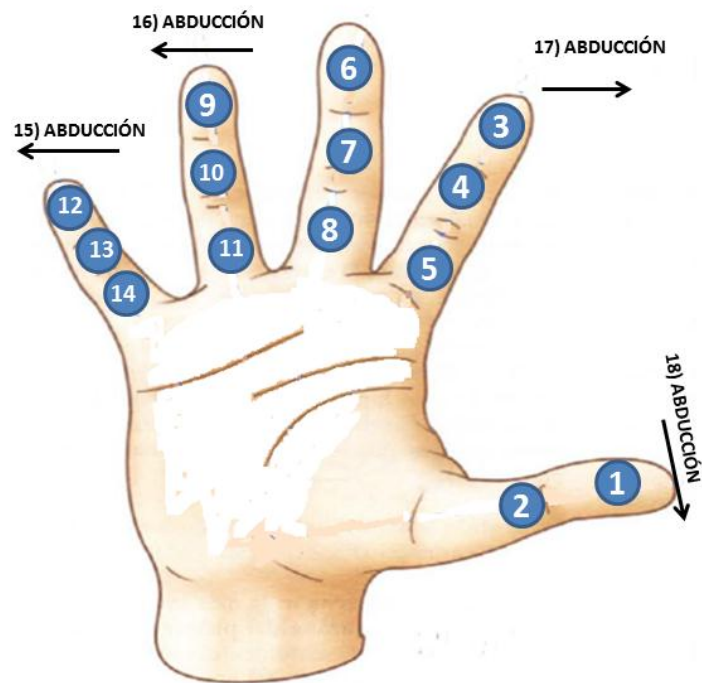


Cap.1: Fig. 36 Oposición subterminolateral.

FUENTE: (15)

1.5.8 GRADOS DE LIBERTAD

Cada dedo de la mano humana cuenta con tres falanges: falange distal, media y proximal, mientras que el pulgar solamente tiene la falange distal y proximal. Esto permite que la mano tenga un alto número de grados de libertad, permitiéndole múltiples configuraciones de aprensión y manipulación, por poseer articulaciones tipo bisagra.



Cap.1: Fig. 37 Ejemplo de una mano con 18 grados de libertad

FUENTE: AUTOR.

CAPÍTULO 2

2. FUNDAMENTOS TEÓRICOS

Durante muchos siglos se intentó encontrar leyes fundamentales que se apliquen a todas o por lo menos a la mayoría de experiencias cotidianas concernientes al movimiento. Siendo en la época de Galileo y Newton donde se efectuaron progresos en lo que se refiere a la inercia de los cuerpos.

El propósito de este capítulo es el de poner los fundamentos necesarios para el desarrollo del “Guante Electrónico de Datos” e incorporar el conocimiento del principio de inercia para los Sistemas de Navegación Inercial (INS) con cada uno de sus componentes.

2.1 PRINCIPIO DE INERCIA

El termino inercia fue introducido por Kepler en el siglo XVII, y definía a la inercia como la tendencia que tienen los cuerpos al estado de reposo, es decir todo cuerpo en movimiento tiene un único fin que es el estado de reposo o velocidad cero.

Galileo en el mismo siglo profundizó e influyó en la teoría de la dinámica de los cuerpos a través de la demostración del movimiento de una esfera sobre un plano inclinado; señalando de esta manera que la inercia es la tendencia de los cuerpos a un estado de movimiento permanente y no a un estado de reposo como proponía Kepler; planteando así, las bases de lo que luego Newton definiría como “principio de inercia”, o primera ley de Newton⁸.

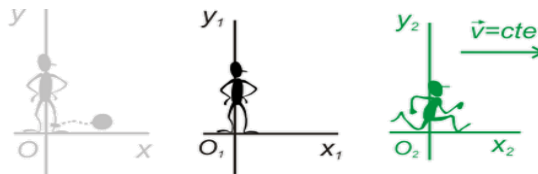
⁸ La ley de Newton sobre la inercia, dice que si sobre un cuerpo no actúa ninguna fuerza, este permanecerá indefinidamente moviéndose en línea recta con velocidad constante (incluido el estado de reposo, que equivale a velocidad cero).

▪ INERCIA

Es la propensión de los cuerpos a mantener constante su velocidad de rotación y de traslación a no ser que se vean afectados por fuerzas o momentos, es decir si una partícula no experimenta una fuerza externa seguirá con una velocidad que llevará de forma constante y si está en un estado de reposo no experimentará movimiento si no se le aplica una fuerza externa.

▪ SISTEMA DE REFERENCIA INERCIAL

Un sistema de referencia inercial es un conjunto de coordenadas espacio-tiempo que se requiere para determinar la posición de un punto en el espacio dentro de un sistema que está en reposo o con movimiento rectilíneo uniforme, respecto de un objeto material sobre el cual no actúa fuerza alguna (21).



Cap.2: Fig. 1 Sistema de referencia inercial.
FUENTE (21).

El sistema de referencia inercial se localiza en el observador (Fig. 1), éste no posee aceleración O_1 (en reposo) o cuando su movimiento es con velocidad constante observador O_2 . Los sistemas de referencia inercial ni rotan ni aceleran.

2.2 SISTEMA DE NAVEGACIÓN INERCIAL (INS)

Un INS es un conjunto de sensores empleados para obtener la aceleración en cada uno de los tres ejes de movimiento X, Y y Z de esta forma basados en los sensores de movimiento (acelerómetros), de rotación (giroscopios) y un pequeño procesador, un INS es capaz de estimar la posición, orientación y velocidad de un objeto. Se utilizan en navegación marítima, aeronaves, misiles y naves espaciales.

Un INS es capaz de detectar un cambio en la posición geográfica (desplazamiento al norte, sur, este, oeste), un cambio en su velocidad (módulo de dirección), un

cambio en su orientación (rotación alrededor de un eje); sólo necesita una referencia externa al inicio.

Los sistemas de navegación inercial cumplen las siguientes funciones:

- Facilitan un marco de referencia para el movimiento del móvil en el espacio.
- Miden una fuerza específica.
- Tienen conocimiento del campo gravitacional de la tierra.
- Ejecutan la integración en el tiempo de la fuerza específica, obteniendo velocidad y posición.

El sistema vestibular, situado en el oído interno, es un sistema biológico de sensores inerciales que está relacionado con el equilibrio y el control espacial. Este detecta el movimiento angular, así como la aceleración lineal de la cabeza. Este sistema ayuda a mantener el equilibrio y estabilidad de los ojos en relación con el medio ambiente.

2.2.1 COMPONENTES BÁSICOS DE UN INS

Los dispositivos que facilitan la orientación, velocidad y posición de un objeto en un INS son sensores que basan su funcionamiento en movimiento y rotación, su aplicación depende de la utilidad que brindan cada uno por separado o en conjunto.

▪ ACELERÓMETRO

Miden aceleración lineal en un entorno fijo no rotatorio. No censan la presencia de un campo gravitacional sino que registran reacciones de fuerzas gravitacionales, necesita de una referencia externa para obtener información de navegación.

▪ GIROSCOPIO

Mide el movimiento de rotación inercial. La orientación inicial esta dada por: razón angular, incrementos o desplazamientos angulares. Muestran características del momento angular.

2.2.2 CONCEPTOS BÁSICOS DE NAVEGACIÓN INERCIAL

▪ SENSORES INERCIALES

Componente principal de los INS, miden e informan acerca de la velocidad, orientación, fuerzas gravitacionales variación de rotación (giroscopio) y aceleración (acelerómetros).

▪ NAVEGACIÓN

Actividad que permite determinar la posición, velocidad y orientación de un vehículo en función del tiempo, con respecto a uno o varios sistemas de referencia seleccionados.

▪ NAVEGACIÓN INERCIAL

Navegación basada en instrumentos inerciales (IMU) Inertial Measurement Unit. Determina la posición, velocidad y actitud actual con la mayor precisión posible a partir de la información que aportan los componentes de una IMU (acelerómetros y giroscopios).

▪ ACTITUD

Es la orientación o referencia angular de los ejes longitudinal y transversal con respecto al horizonte. La actitud devuelve la medida del ángulo de inclinación con respecto a los ejes de referencia que se usan, dada por los ángulos de Euler (roll, pitch y yaw). El dispositivo usado para verificar la actitud en sistemas de navegación aérea es el AHRS (Attitude and Heading Reference System) u Horizonte Artificial.

▪ IMU

Dispositivo electrónico principal de los INS, usados en aviones, barcos, etc. que incorpora sensores inerciales (acelerómetros y giroscopios) utilizados para el estudio y análisis del movimiento, en base a las variables de aceleración y velocidad angular de un aparato. La ventaja es: la medición casi instantánea en la aceleración y el giro,

la desventaja es que acumula un error denominado error de deriva. Para resolver esto, el sistema cuenta con un método de autocalibración.

▪ **EQUILIBRIO**

Un móvil se encuentra en equilibrio, si la suma de todas las fuerzas y momentos en su centro de gravedad es igual a cero (reposo). También con esta condición el móvil podría moverse con velocidad constante.

▪ **ESTABILIDAD**

Es la capacidad de un móvil para mantener el equilibrio y recuperarse de los efectos de condiciones perturbadoras o movimientos bruscos, si el móvil no se mantiene en equilibrio luego de aplicarse una fuerza externa y cambia de posición se habla de una estabilidad negativa.

▪ **CONTROL**

Es la capacidad de respuesta de un móvil en diferentes condiciones de uso, midiendo el valor existente y comparándole con el valor deseado; utilizando la diferencia para proceder a reducirla.

2.2.3 SISTEMA DE COORDENADAS

El sistema de coordenadas es un conjunto de vectores y números que permiten determinar la posición de un punto en el espacio; adquieren un significado cuando se los relaciona con una referencia conocida, en este caso su relación de referencia es con la tierra (gravedad).

2.2.3.1 EJES DE MOVIMIENTO DE NAVEGACIÓN INERCIAL

Se refiere a las rectas imaginarias trazadas sobre un cuerpo en el espacio. Su denominación y los movimientos que se ejecutan alrededor de cada eje. Los ejes de movimiento para la navegación inercial son:

▪ EJE LONGITUDINAL (X)

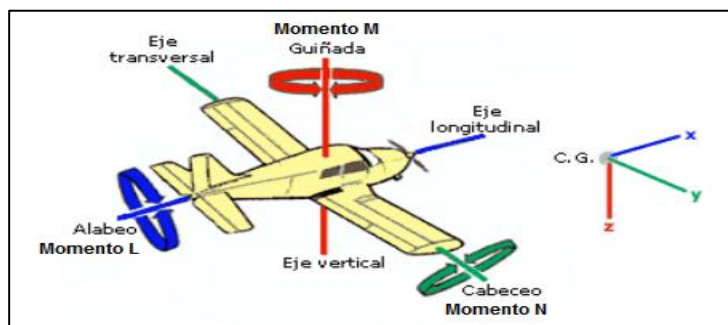
El origen está en el centro de gravedad situado en el plano de simetría del móvil. Alrededor de este eje se produce un movimiento de balanceo o alabeo (roll) momento L.

▪ EJE TRANSVERSAL O LATERAL (Y)

El origen está en el centro de gravedad y es perpendicular al plano de simetría del móvil siendo su sentido positivo hacia abajo. Alrededor de este eje se produce un movimiento de cabeceo (pitch), momento N.

▪ EJE VERTICAL (Z)

El origen está en el centro de gravedad, está situado en el plano de simetría del móvil, su sentido positivo hacia la derecha. Alrededor de este eje se produce un movimiento de guiñada (yaw), momento M.



Cap.2: Fig. 2 Ejes de movimiento de navegación inercial.

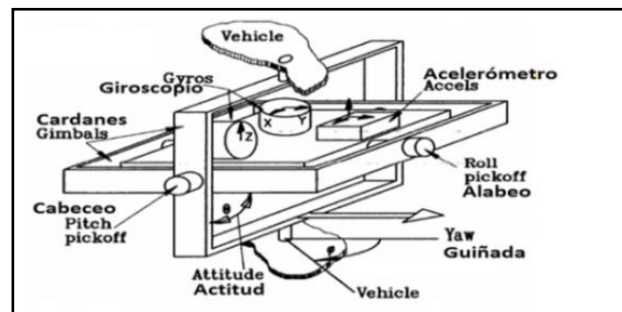
FUENTE: (22)

2.2.4 TIPOS DE SISTEMA DE NAVEGACIÓN INERCIAL

En la segunda guerra mundial se emplearon los sistemas de navegación inercial para guiar misiles, luego de la guerra hubo un avance considerable con respecto a la navegación inercial, que consistía en un conjunto de acelerómetros y giroscopios mecánicos cuyas desventajas principales era el alto costo, el tamaño y desgaste de sus elementos. En la actualidad se emplean sistemas más compactos es decir integrados en una tarjeta electrónica IMU.

2.2.4.1 SISTEMA DE NAVEGACIÓN INERCIAL GIMBAL (Plataforma inercial)

Su funcionamiento está basado en sistemas mecánicos que unidos a un marco rígido forman una plataforma inercial. Esta plataforma aísla el sistema interno de los movimientos y rotaciones externas; manteniendo fija la parte interior, permitiendo que el exterior continúe rotando en (x, y, z) . De esta manera devuelve información sobre la actitud del móvil.

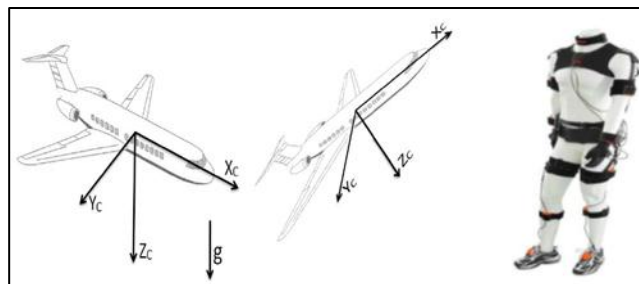


Cap.2: Fig. 3 Componentes principales presentes en una plataforma inercial.

FUENTE: (23)

2.2.4.2 SISTEMA DE NAVEGACIÓN INERCIAL STRAP-DOWN

Denominado también Sistema o plataforma Analítica, basado en sensores inerciales IMU cuya característica principal es que se encuentran unidos al móvil. Usa gran cantidad de recursos computacionales en tiempo real para el cálculo numérico. Son aptos para móviles sometidos a altas aceleraciones y bruscos cambios de actitud como por ejemplo en aviones, en misiles guiados, trajes para juego, etc.



Cap.2: Fig. 4 Sistema strap-down o plataforma analítica

FUENTE: (24)

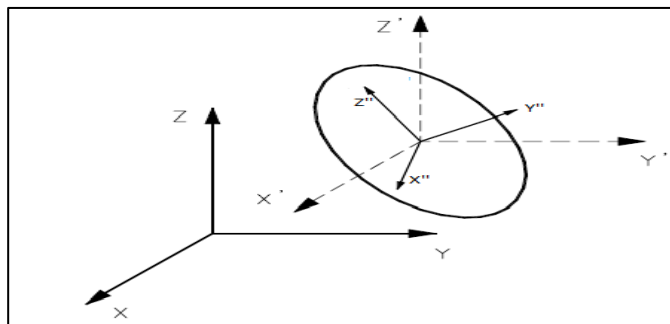
2.2.5 ALGORITMOS DE NAVEGACIÓN

Los valores de aceleración y velocidad angular que censa la IMU son medidas descritas al sistema de coordenadas fijada al móvil (X', Y', Z'). Este movimiento que efectúa el móvil debe basarse a un sistema de referencia inercial (X, Y, Z).

2.2.5.1 GRADOS DE LIBERTAD

Un cuerpo en el espacio tiene seis grados de libertad, por tal motivo se necesitan seis coordenadas independientes para determinar su ubicación. De las cuales, tres especifican los movimientos de traslación (X, Y, Z), y las otras tres determinan los giros (Φ , θ , Ψ). Estas coordenadas determinan la ubicación de un sistema de coordenadas cartesianas que está fijado al cuerpo, respecto a un sistema de coordenadas global que se encuentra fuera del cuerpo **(25)**.

- Sistema global o de referencia fijo (X, Y, Z) localizado en el exterior.
- Sistema local o móvil (X'', Y'', Z'') que acompaña al cuerpo en su movimiento.



Cap.2: Fig. 5 Sistema de referencia global (X, Y, Z) y móvil (X'', Y'', Z'').

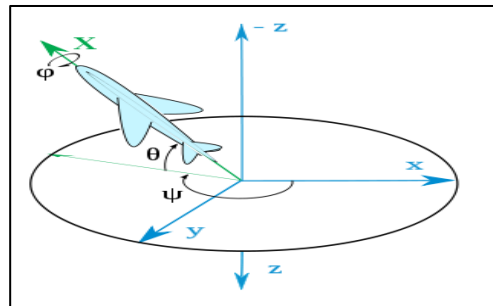
FUENTE: (25)

2.2.5.2 ÁNGULOS DE EULER

Constituyen un conjunto de tres coordenadas angulares que sirven para especificar la orientación de un sistema de referencia de ejes ortogonales, que mediante una sucesión ordenada de giros definen el cambio de un sistema de coordenadas a otro.

Los ángulos de Euler definidos como (Φ , θ , Ψ) que corresponden a los movimientos de:

- eje longitudinal (x) con movimiento de alabeo roll Φ .
- eje transversal (y) con movimiento de cabeceo pitch θ .
- eje vertical (z) con movimiento de guiñada yaw Ψ .



Cap.2: Fig. 6 Movimientos de un móvil en ángulos de Euler.

FUENTE: (26)

2.2.5.2.1 MATRIZ DE ROTACIÓN

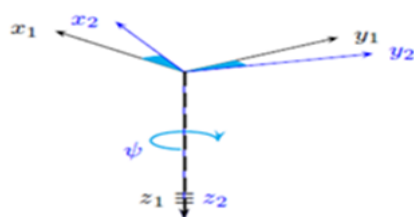
Las matrices de rotación son el método más extendido para la descripción de orientaciones de un móvil. Las tres rotaciones sucesivas que definen la orientación final del móvil se ejecutan respecto a diferentes ejes, por ejemplo: (X, Y, X), o (X', Y', Z'), existiendo un total de 12 combinaciones posibles. En este caso únicamente se tratará con la combinación de Tait Bryan (roll, pitch y yaw) que son usadas en sistemas de navegación. Es preciso acotar que las rotaciones en tres dimensiones no conmutan, por tal motivo si se invierte la secuencia de rotación con otras combinaciones diferentes se obtienen otros resultados.

▪ ROTACIÓN ROLL, PITCH, YAW

Este tipo de representación se utiliza en aeronáutica. Aplicada generalmente en giros sobre los ejes del sistema fijo. Si se parte de los sistemas O,Z,Y,X.

Giro en Yaw

Rotación alrededor del eje Z con ángulo ψ . Con esta matriz de cambio se logra pasar del sistema de coordenadas (X1, Y1, Z1) al nuevo sistema (X2, Y2, Z2).



MATRIZ PARA GIRO EN YAW

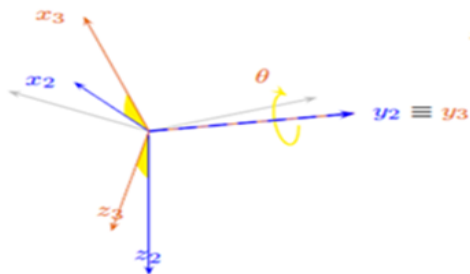
$$A_{\psi} = \begin{bmatrix} \cos \psi & -\sin \psi & 0 \\ \sin \psi & \cos \psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Cap.2: Fig. 7 Giro en Yaw

FUENTE: (27)

Giro en Pitch

Rotación alrededor del eje Y con ángulo θ . El sistema gira $(X2, Y2, Z2)$ para conseguir $(X3, Y3, Z3)$



MATRIZ PARA GIRO EN PITCH

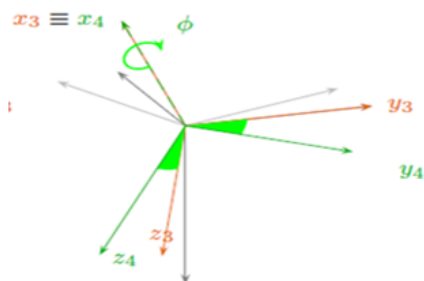
$$A_{\theta} = \begin{bmatrix} \cos \theta & 0 & \sin \theta \\ 0 & 1 & 0 \\ -\sin \theta & 0 & \cos \theta \end{bmatrix}$$

Cap.2: Fig. 8 Giro Pitch

FUENTE: (27)

Giro en Roll

Rotación alrededor del eje X con ángulo ϕ . El sistema gira $(X3, Y3, Z3)$ a $(X4, Y4, Z4)$.



MATRIZ PARA GIRO EN ROLL

$$A_{\phi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \phi & -\sin \phi \\ 0 & \sin \phi & \cos \phi \end{bmatrix}$$

Cap.2: Fig. 9 Giro Roll

FUENTE: (27)

Entonces la matriz de rotación final es:

$$A_{\psi\theta\phi} = \begin{bmatrix} \cos\theta \cos\psi & -\cos\phi \sin\psi + \sin\phi \sin\theta \cos\psi & \sin\phi \sin\psi + \cos\phi \sin\theta \cos\psi \\ \cos\theta \sin\psi & \cos\phi \sin\psi + \sin\phi \sin\theta \sin\psi & -\sin\phi \cos\psi + \cos\phi \sin\theta \cos\psi \\ -\sin\theta & \sin\phi \cos\theta & \cos\phi \cos\theta \end{bmatrix}$$

2.3 CUATERNIONES

En el desarrollo de las matemáticas los cuaterniones fueron el primer ejemplo dentro de un campo no conmutativo y la base del algebra vectorial que se emplea actualmente en la que las unidades matemáticas i, j, k pasaron a ser los vectores i, j, k relacionados con las rotaciones en el espacio tridimensional **(28)**.

Los cuaterniones (cuatro-dimensionales) son miembros de un cuerpo no conmutativo que fueron introducidos por W.R. Hamilton en 1843, que describen puntos en el espacio tridimensional. En forma similar de como los números complejos son una extensión de los números reales por la adición de la unidad imaginaria i tal que $i^2 = -1$, los cuaterniones son una extensión generada de manera análoga añadiendo las unidades imaginarias i, j, k a los números reales tal que:

$$i^2 = j^2 = k^2 = -1.$$

El cuaternión i representa una rotación de 180° alrededor del eje X, j alrededor del eje Y, k alrededor de Z. Concluyendo que $i * i = i^2 = -1$ representan una rotación de 360° alrededor de X en el caso de i^2 .

Por analogía de los números complejos, los cuaterniones se representan como la suma de una parte real y una imaginaria.

$$q = a \cdot 1 + bi + cj + dk$$

Siendo a, b, c, d la parte real o escalar.

La forma matricial de los cuaterniones es la siguiente:

$$q = \begin{bmatrix} z & w \\ -\bar{w} & \bar{z} \end{bmatrix} = \begin{bmatrix} a + ib & c + id \\ -c + id & a - ib \end{bmatrix}$$

Donde Z y W son números complejos a, b, c, d son números reales y \bar{z} es el complejo conjugado de Z .

2.3.1 OPERACIONES CON CUATERNIONES

Las operaciones que se ejecutan con los cuaterniones son similares a las operaciones que se ejecutan con los números complejos.

▪ SUMA

Cada término se suma con su homólogo del otro número.

$$a + b = (a_1 + b_1) + (a_2 + b_2)i + (a_3 + b_3)j + (a_4 + b_4)k$$

▪ CONJUGADO

Conocido un número complejo su conjugado se obtiene cambiando el signo de la parte imaginaria.

$$q = a \cdot 1 + bi + cj + dk$$

$$q^* = a \cdot 1 - bi - cj - dk$$

Propiedades de la conjugada de los cuaterniones.

$$q^* = \text{conjugado de } q$$

$$(q^*)^* = q$$

$$(pq)^* = q^*p^*$$

$$(p+q)^* = p^*+q^*$$

▪ MULTIPLICACIÓN

El producto de cuaterniones no es conmutativo y se resume mediante la tabla de multiplicación de Cayley. En la columna siempre se coloca los elementos del primer cuaternión.

q1/q2	1	i	j	k
1	1	i	j	k
i	i	-1	k	-j
j	j	-k	-1	i
k	k	j	-i	-j

PRODUCTOS PRINCIPALES:

$$i \times j = k \quad j \times i = -k$$

$$j \times k = i \quad k \times j = -i$$

$$k \times i = j \quad i \times k = -j$$

Cap.2: Tabla. 1 Tabla de Multiplicación de Cayley para cuaterniones.

FUENTE: (29)

▪ LA NORMA

La norma es la raíz cuadrada de la suma de sus cuatro componentes al cuadrado. La norma o valor absoluto viene dada por un número real no negativo, cuando es 1 se dice que el cuaternión está normalizado.

$$q_1 = a + a_1 i + a_2 j + a_3 k$$

$$|q| = \sqrt{a^2 + a_1^2 + a_2^2 + a_3^2}$$

▪ INVERSO

Esta definido por el conjugado dividido por la norma al cuadrado.

$$q^{-1} = \frac{q^*}{\|q\|^2}$$

2.3.2 REPRESENTACIÓN DE PUNTOS A TRAVÉS DE CUATERNIONES

Todo punto del espacio tridimensional se representa mediante un cuaternión cuya parte real es cero.

$$\text{Si } P = (P_1, P_2, P_3) \text{ entonces } P = 0 + P_1 i + P_2 j + P_3 k$$

2.3.3 REPRESENTACIÓN DE VECTORES A TRAVÉS DE CUATERNIONES

Todo vector del espacio tridimensional es representado mediante un cuaternión cuya parte real es cero.

$$\text{Si } V = V_1i + V_2j + V_3k \quad \text{entonces } V = 0 + V_1i + V_2j + V_3k$$

2.3.4 CUATERNIONES UNITARIOS

Es aquel cuaternión cuya norma o valor absoluto es igual a uno $a^2 + a_1^2 + a_2^2 + a_3^2 = 1$. La importancia de los cuaterniones unitarios reside en que a través de ellos se representan rotaciones en tres dimensiones de manera muy sencilla y evita el problema del gimbal lock. Si q es un cuaternión unitario se representa como una esfera de radio 1 en el espacio 4D. De tal manera que se representa una rotación en el espacio 4D, donde (a_1, a_2, a_3) son los componentes de cualquier eje arbitrario y a es el ángulo de rotación.

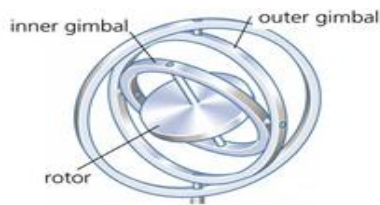
Un cuaternión unitario es representado por:

$$q = \cos \theta + u \sin \theta$$

Donde u es un vector de 3D de longitud 1.

2.3.5 GIMBAL LOCK

Anteriormente se mencionó al gimbal como un tipo de sistema de navegación inercial, aquí se profundiza este tema ya que es de mucha importancia para poder entender la utilidad de los cuaterniones en lo que se refiere a la navegación espacial.



Cap.2: Fig. 10 Modelo básico de una plataforma con gimbals

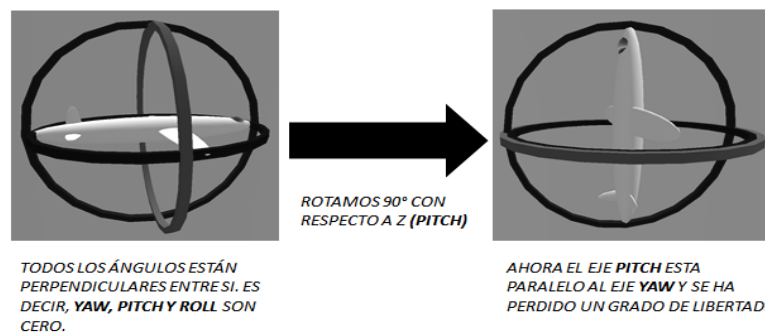
FUENTE: (30).

El gimbal o cardán cuyo nombre proviene de su inventor Girolamo Cardano, es parte importante del giroscopio, cuyo soporte pivoteado permite la rotación de un anillo alrededor de un eje, de tal manera que el objeto mantiene su orientación en el espacio sin importar con qué ángulo se encuentre la base o estructura.

Generalmente estos anillos se anidan uno dentro del otro produciendo de esta manera rotaciones en ejes diferentes.

A partir de lo anterior se dice que el gimbal lock (bloqueo del cardán) es la pérdida de un grado de libertad que ocurre cuando dos de los tres ejes necesarios para las rotaciones en el espacio tridimensional son llevados a la misma dirección, por lo tanto no existe un eje de rotación disponible para rotar en la dirección que falta; dando la apariencia de tener rotaciones en un espacio de dos dimensiones.

Un incidente de “Gimbal Lock” bien conocido sucedió en la misión lunar “Apolo 11” en la cual usaron un conjunto de cojinetes en una IMU. Fue necesario mover manualmente la nave para sacarla de la posición de “Gimbal Lock”



Cap.2: Fig. 11 Ejemplo de gimbal lock

FUENTE: (31)

Una solución presentada para el gimbal lock es el uso de un cuarto gimbal movido por medio de un motor. Otra manera es la de reiniciar el dispositivo, previamente moviendo de manera arbitraria un gimbal. Estas soluciones se ejecutan en el caso de un dispositivo mecánico; pero para un dispositivo MEMS este problema es resuelto mediante el uso de los cuaterniones. Las rotaciones con cuaterniones son definidas con los 4 términos de éste; si se produce un bloqueo, los 4 términos no le permiten bajar de los 3 grados de libertad.

2.3.6 ROTACIONES

En 1845 Arthur Cayley publica la manera de describir rotaciones usando la multiplicación de los cuaterniones.

Los cuaterniones tienen la capacidad de capturar toda la geometría, topología, y estructura de las rotaciones tridimensionales en la forma más sencilla posible.

Para representar una rotación alrededor de un eje específico mediante cuaterniones, se obtiene mediante un par ordenado compuesto por una parte escalar y una parte vectorial.

$$q = [w, v]$$

$$\text{Donde } v = [x, y, z] \text{ y } w = s$$

La parte vectorial v es la que representa la rotación del cuerpo en esa dirección con respecto a un ángulo s dado. De tal manera que en un cuaternión de rotación se define con los siguientes términos:

Escalares: $s = \cos \frac{\theta}{2}$

Vectoriales: $v = u \sin \frac{\theta}{2}$

Entonces: $q = \cos \frac{\theta}{2}, u \sin \frac{\theta}{2}$

En la parte vectorial: u es un vector unitario a lo largo del eje de rotación y θ es el ángulo de rotación específico con respecto del eje seleccionado. A partir de esto, cualquier punto P que se desee rotar mediante un cuaternión se representa como:

$$P = (0, p)$$

Las coordenadas de P son la parte vectorial (x, y, z) . Siendo la rotación de la siguiente manera:

$$P' = qPq^{-1}$$

$$P' = q^* [0, p]^* q^{-1}$$

Siendo $q^{-1} = (s, -v)$, el inverso del cuaternion de rotación q .

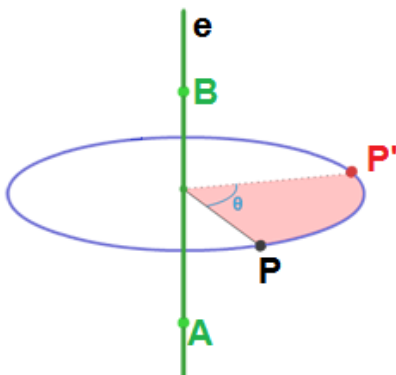
Esto produce un cuaternión con la parte escalar cero, donde la parte vectorial es la rotación de p con respecto a la dirección v y un ángulo θ .

También se define una rotación por medio de una matriz para los cuaterniones, Siendo $q = (s, a, b, c)$.

$$M_r(\theta) = \begin{bmatrix} 1 - 2b^2 - 2c^2 & 2ab - 2sc & 2ac + 2sb \\ 2ab + 2sc & 1 - 2a^2 - 2c^2 & 2bc - 2sa \\ 2ac - 2sb & 2bc + 2sa & 1 - 2a^2 - 2b^2 \end{bmatrix}$$

2.3.6.1 DEMOSTRACIÓN DE UNA ROTACIÓN CON CUATERNIONES

El caso más general de rotaciones con cuaterniones con respecto a cualquier eje arbitrario en el espacio consiste en la rotación de un punto $P(px, py, pz)$ alrededor de un eje arbitrario definido por dos puntos $A(ax, ay, az)$ y $B(bx, by, bz)$ **(32)**.



Cap.2: Fig. 12 Demostración de rotación con cuaterniones.

FUENTE: (32)

En la Fig.12 se rotará un punto P en el espacio tridimensional, para obtener la posición final en el punto P', luego de una rotación de theta radianes. Para esto se procede de la siguiente manera:

Se convierte el punto $P(px, py, pz)$ en un cuaternión:

$$Qp = (0, px, py, pz).$$

En la recta A y B se obtienen el eje de rotación e a través de una resta vectorial $e = B - A$ y se normaliza el resultado con $(ex/|e|, ey/|e|, ez/|e|)$.

Con el eje de rotación e normalizado y el ángulo de rotación theta (θ) se construye el cuaternión de rotación:

$$Q_R = \cos(\theta/2) + \sin(\theta/2).$$

De esta manera se obtiene el cuaternión Q_R con los siguientes componentes:

$$w = \cos(\theta/2), \quad x = \text{exn} \sin(\theta/2), \quad y = \text{eyn} \sin(\theta/2), \quad z = \text{ezn} \sin(\theta/2).$$

Donde exn , eyn , ezn son los vectores unitarios a lo largo del eje de rotación e .

El resultado de la rotación P respecto al eje e se representa por el producto de cuaterniones:

$$Q_{P'} = Q_R Q_P Q_{R^{-1}}$$

Y finalmente se añade el punto A , de esta manera se gira el punto con respecto al eje AB .

$$Q_{P'} = A + Q_R Q_P Q_{R^{-1}}$$

La componente escalar w de $Q_{P'}$ tendrá siempre un valor 0 obteniendo directamente el nuevo punto P' .

2.3.7 APLICACIONES

- En electromagnetismo y mecánica cuántica.
- Para gráficos por computadora.
- En representación de orientación de un objeto en un espacio tridimensional.

2.3.8 VENTAJAS

- La representación de una rotación es más compacta.
- Permite hacer rotaciones suaves.

- No existe el gimbal lock.

2.3.9 RENDIMIENTO

Método	Almacenaje	Multiplicaciones	Sumas/Restas	Total
Matrices de Rotación	9	27	18	45
Cuaternión	4	16	12	28

Cap.2: Tabla. 2 Comparación de rendimiento por operaciones.

FUENTE: (31)

2.4 TECNOLOGÍA MEMS (Sistemas Micro-Electro-Mecánicos)

Los MEMS son dispositivos de bajo consumo fabricados en una base de silicio, comprende la integración de elementos electrónicos y mecánicos mediante el empleo de la microelectrónica y las tecnologías de micromaquinado sobre un mismo sustrato.

Los MEMS miden variables térmicas, eléctricas, magnéticas, biológicas y devuelven una acción en un dispositivo mecánico. Su uso dentro del área computacional, comunicaciones, control de micro-sensores y micro-actuadores permite el desarrollo de sistemas inteligentes con capacidades de sensado, procesamiento, comunicación y actuación en un mismo circuito, siendo su tamaño de escala desde las décimas de micra hasta los cientos de micras.

Las primeras exploraciones sobre tecnologías MEMS se empezaron en la década de los 60s, gracias a la invención previa del transistor y del microprocesador (1950), siendo en los años 90 cuando se observó los primeros avances en lo que a comercialización y utilización de los MEMS se refieren.

2.4.1 TIPOS DE MEMS

Dentro de los MEMS existen sensores y actuadores que ejecutan una determinada función electromecánica o electroquímica, se tiene por ejemplo: microválvulas, microbombas, y microengranajes.

2.4.1.1 MICROACTUADORES

Son aquellos que responden a la información, convirtiendo una señal eléctrica en una magnitud no eléctrica como un desplazamiento, por ejemplo: microinterruptores, microválvulas, microelectrodos que se usan para estimular tejido nervioso en aplicaciones de prótesis.

2.4.1.2 MICROSENSORES

Aquellos que detectan la información, convierte una magnitud de entrada no eléctrica (presión, temperatura, aceleración, luz) en una señal eléctrica como salida. Se le conoce también como sensor inteligente porque uno de sus elementos es un procesador digital que le proporciona capacidad de autocalibración, autodiagnóstico y comunicaciones.

2.4.2 COMPONENTES MEMS

Los componentes de los MEMS están formados por varios elementos fabricados en una base de silicio, estos elementos son: diafragmas, vigas, engranajes, componentes de control de fluido como boquillas y conductos.

Según su aplicación se logran integrar los siguientes elementos: sensores, actuadores, cómputo, energía, comunicación.



Cap.2: Fig. 13 Funcionamiento general de los MEMS.

FUENTE: (33).

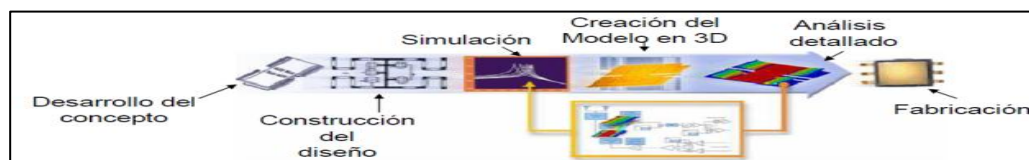
2.4.3 CONSTRUCCIÓN DE LOS MEMS

Las propiedades de los materiales con los cuales se construyen los MEMS definen su rendimiento y durabilidad. Los materiales que se usan son plástico, cuarzo diamante, siendo el mayormente usado el silicio por sus propiedades físicas, mecánicas y funcionalidad electrónica. El sustrato es un objeto plano macroscópico sobre el cual se efectúa el proceso de microfabricación. Adicionalmente el sustrato actúa como transductor de la señal o ayuda a otros transductores a transformar una magnitud física de entrada en una señal eléctrica como salidas o viceversa.

TÉCNICAS DE FABRICACIÓN

Las tres técnicas más utilizadas en la fabricación de MEMS son LIGA (Litografía, Electrodeposición y Moldeo), micro-maquinado superficial y micro-maquinado volumétrico.

Los MEMS son construidos usando partes producidas con diferentes tecnologías. Por ejemplo, un circuito de silicio usado como circuito de control, los actuadores que controlan este circuito generalmente son moldeados en plástico o metal galvanizado usando la técnica LIGA.



Cap.2: Fig. 14 Proceso general de fabricación de los MEMS

FUENTE: (33)

Un problema significativo que enfrentan los micro-maquinados es el ensamblado de muchos componentes microscópicos. Las soluciones que se sugieren son: sistemas de auto-ensamblado y fábricas controladas por micro-robots.

2.4.4 VENTAJAS

- Reducción del consumo de potencia.
- Dispositivos más veloces.
- Aumento de selectividad y la sensibilidad.
- Integración con electrónica, simplifica los sistemas.
- Larga vida del producto.

2.4.5 APLICACIONES

- Automotriz, sensores térmicos acústicos.
- Aeroespacial, sensores de vibración, IMU.
- Industria, detectores de gas, radiación.
- Construcción, sensores químicos y de presión.

2.5 DISPOSITIVOS INERCIALES

Los dispositivos inerciales más conocidos son el acelerómetro y el giroscopio, trabajan individualmente o también en conjunto cuando son parte de una IMU.

2.5.1 EL ACELERÓMETRO

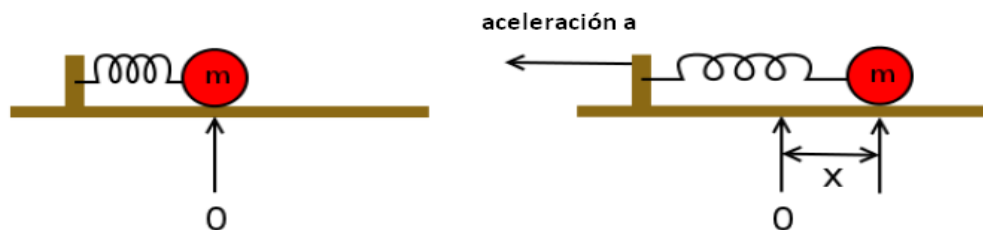
El acelerómetro es un dispositivo que permite medir aceleraciones propias de un sistema; estas aceleraciones están asociadas con el fenómeno de peso experimentado por una masa de prueba que se encuentra en el marco de referencia del dispositivo. Existen dos tipos diferentes de aceleraciones: aceleración estática

(inclinación) y la aceleración dinámica (movimiento), la salida de un acelerómetro provee una medición que indica al usuario en que dirección está siendo aplicada una fuerza.

Los acelerómetros miden la aceleración en unidades “ g ”, que se define como la fuerza gravitacional de la tierra aplicada sobre un objeto o persona.

2.5.1.1 FUNCIONAMIENTO DE UN ACELERÓMETRO

El funcionamiento básico de un acelerómetro es entendido mediante la idea de un sistema masa–resorte, sin embargo en la actualidad no están constituidos de esta forma, Fig.15 permite hacer una aproximación a este sensor.



Cap.2: Fig. 15 Funcionamiento General de un acelerómetro.

FUENTE: (34).

Cuando el sistema permanece estable y no experimenta ningún tipo de aceleración, la masa se encuentra en el punto cero. Cuando al sistema se aplica una aceleración (a), la masa se desplaza una distancia (x) con respecto al origen cero.

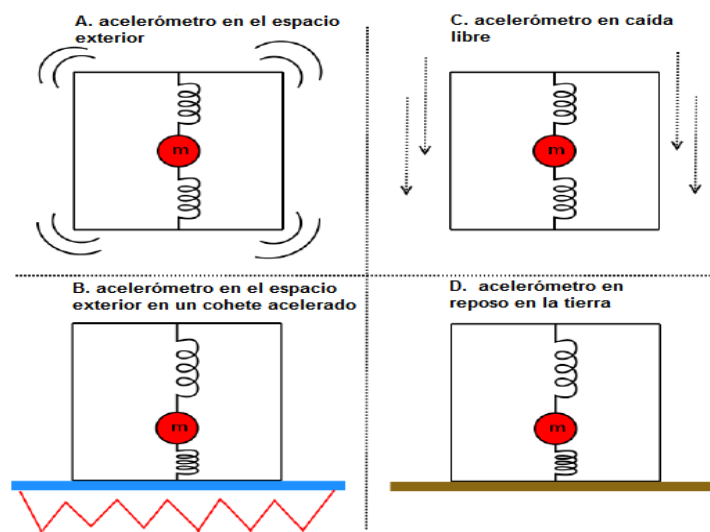
Por la segunda ley de Newton para el movimiento se tiene que; una masa (m) que es acelerada por una aceleración (a) estará sometida a una fuerza $F = ma$, como la masa está también conectada a un resorte, según la ley de Hooke, esto genera una fuerza (F) opuesta y proporcional a (x), de manera que $F = kx$ donde (k) es la constante del resorte y su valor dependerá de las características del mismo.

Como el movimiento de la masa siempre estará limitado por el resorte, se tiene que $F = ma = kx$. Conociendo los valores de (k , m , x) se calcula la aceleración externa usando simplemente $a = kx/m$ (34).

Cuando al acelerómetro se le aplica una fuerza, la masa tiende a oponerse al movimiento debido a su propia inercia, esto produce un desplazamiento de la masa; este desplazamiento es medido mediante un circuito eléctrico.

2.5.1.2 EL ACELERÓMETRO Y LA GRAVEDAD

Como el acelerómetro es un dispositivo que sirve para medir aceleraciones propias del dispositivo, a continuación se describe algunos ejemplos que permiten entender lo que en realidad esto significa.



Cap.2: Fig. 16 Aceleraciones externas y efectos de la gravedad.

FUENTE: (34).

Considerando que el acelerómetro está en el espacio exterior (Fig.16.A). Si el dispositivo está en reposo, no hay fuerzas o aceleraciones que actúan sobre el sensor y su masa interna, por lo tanto en su salida se leerá un valor igual a cero.

Si el sensor ahora se pone en una nave espacial (Fig.16.B), se genera una aceleración (a) producida por los motores, la masa experimenta una fuerza $F = ma$ contrastada por la fuerza elástica del resorte $F = kx$ de manera que se infiere que (a) tiene un valor diferente de cero.

Por otro lado, si se deja caer al sensor libremente bajo el efecto de la gravedad y en ausencia de la fricción causada por el aire (Fig.16.C), todo el acelerómetro se acelera bajo efectos de una aceleración (g) donde $g = 9.81 \text{ m/s}^2$. Tal situación se

denomina caída libre, los objetos parecen no tener peso, de manera que la masa en el interior del dispositivo no provoca desplazamiento alguno a los resortes y en la salida del sensor se leerá una aceleración igual a 0. Esto es porque según la ley de Newton manifiesta que un cuerpo en caída libre es un sistema inercial tal que la suma de las fuerzas gravitacionales e inerciales es igual a cero.

Si se coloca el acelerómetro en estado de reposo en la Tierra (Fig.16.D), el dispositivo leerá un valor de $a = g$. Esto es porque el peso de la masa (m) será sometido a la fuerza de gravedad, resultando un desplazamiento de la masa hacia abajo. En estas condiciones el dispositivo lo que realmente mide es la fuerza normal F_n que ejerce el suelo sobre el acelerómetro.

Con lo dicho anteriormente se demuestra que un acelerómetro está sujeto al efecto de la gravedad:

- Cuando se pone en reposo sobre la Tierra, se lee una aceleración $a = g$.
- Durante la caída libre del sensor, se lee una aceleración $a = 0$.
- Si se desea obtener las coordenadas de la aceleración (cambio de la velocidad del dispositivo en el espacio), se debe remover o restar la gravedad de las salidas del acelerómetro, haciendo lo que se denomina compensación de la gravedad.
- Si se rota el acelerómetro, el efecto de la gravedad en su masa interna variará con el ángulo de rotación, dando una salida diferente con diferente ángulo de rotación. Se usa esta característica para poner en práctica la detección de inclinación con este sensor.

2.5.1.3 TIPOS DE ACELERÓMETRO

A) ACELERÓMETRO PARA MEDIDAS ESTÁTICAS

Son acelerómetros que ofrecen respuestas ante aceleraciones que no varían en el tiempo o ante aceleraciones de muy baja frecuencia o frecuencia nula. Un ejemplo de este tipo de aceleraciones es la aceleración de la gravedad.

B) ACELERÓMETROS PARA MEDIDAS DINÁMICAS

Los acelerómetros para medidas dinámicas están preparados para medir señales de aceleración que varían rápidamente en el tiempo; clasificándose en dos tipos.

- **ACELERACIÓN DINÁMICA PERSISTENTE** También llamada de vibración, cuyas aceleraciones se prolongan en el tiempo.
- **ACELERACIÓN DINÁMICA TRANSITORIA** Suceden en un espacio de tiempo muy breve y suele ser provocada por impacto.

C) ACELERÓMETROS PARA LA MEDICIÓN DE UNA O VARIAS DIMENSIONES

Usados para medir un vector de aceleración cuya dirección no es completamente conocida o varía en el tiempo de manera arbitraria.

Los acelerómetros son direccionales, esto quiere decir que solo miden aceleración en un eje. Para monitorear aceleración en tres dimensiones, se emplean acelerómetros multi-ejes (x,y,z), estos son ortogonales; es decir son independientes y se necesitan entre si; para determinar la orientación en el espacio.

- **ACELERÓMETRO MONOAXIAL**

Posee un elemento sensor que mide la aceleración paralela a un solo eje de actuación (x, y, ó z), es utilizado en el campo de la instrumentación.

- **ACELERÓMETRO BIAXIAL**

Posee dos sensores dispuestos de manera perpendicular, lo cual permite medir la aceleración en dos ejes de coordenadas. Utilizados para medir aceleración que se genera en un plano.

- **ACELERÓMETRO TRIAXIAL**

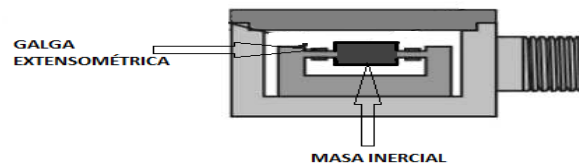
Poseen tres elementos sensores para cada eje (x, y, z). Cada sensor proporciona una señal eléctrica en función de la aceleración del respectivo eje.

2.5.1.4 TECNOLOGÍAS DE FUNCIONAMIENTO DE LOS ACELERÓMETROS

Existen varios tipos de tecnologías y diseños, que aunque todos tienen el mismo fin son muy distintos unos de otros según el trabajo para el cual son destinados.

▪ ACELERÓMETRO MECÁNICO

Utiliza una masa inerte y resortes elásticos con sistema de amortiguación que evitan su propia amortiguación. Los cambios se miden con galgas extensométricas, estos cambios producen una deformación de la galga, que es proporcional a la aceleración aplicada al acelerómetro.

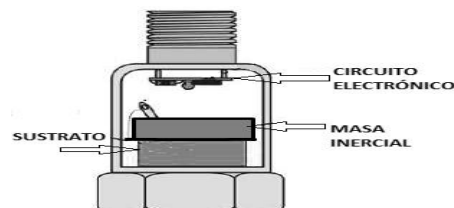


Cap.2: Fig. 17 Acelerómetro mecánico.

FUENTE: (35).

▪ ACELERÓMETROS PIEZORESISTIVOS

Su funcionamiento se basa en el efecto piezo-resistivo. Este efecto cambia el valor de la resistencia eléctrica del puente cuando sucede una deformación física del material. Mide aceleraciones en bajas frecuencias de hasta cero HZ.



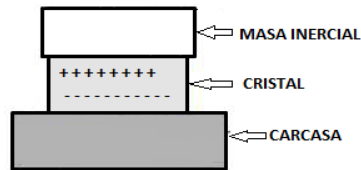
Cap.2: Fig. 18 Acelerómetro piezoresistivo.

FUENTE: (35).

▪ ACELERÓMETROS PIEZOELÉCTRICOS

Su funcionamiento es de efecto piezoeléctrico. La deformación física del material causa un cambio en la estructura cristalina y de esta manera cambia las

características eléctricas. El principal inconveniente es que trabaja a alta frecuencia.

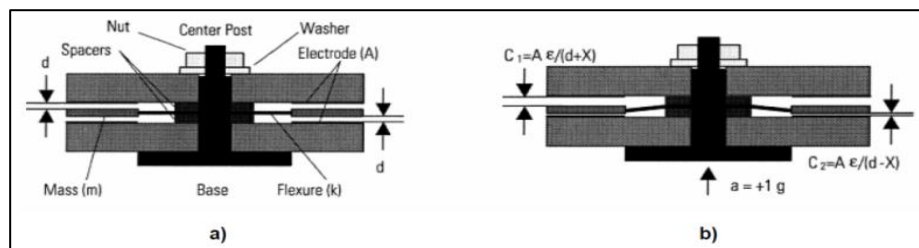


Cap.2: Fig. 19 Acelerómetro piezoeléctrico.

FUENTE: (35)

▪ ACELERÓMETRO CAPACITIVO

Estos acelerómetros basan su funcionamiento en el efecto condensador para transformar una magnitud física en una variación de la capacidad de un condensador que es traducida en una variación de tensión eléctrica. Un acelerómetro capacitivo está constituido por uno o más condensadores que varían su capacidad por efecto de la aceleración por la variación de la separación de las placas.



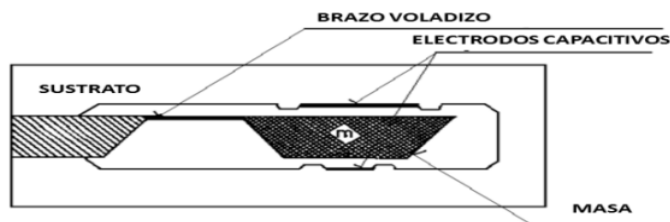
Cap.2: Fig. 20 Acelerómetro capacitivo⁹.

FUENTE: (36).

▪ ACELERÓMETRO MEMS

Acelerómetros micromecánicos miden el movimiento tal como la aceleración, vibración, choque e inclinación, inicialmente fue utilizado en la industria automotriz y aeroespacial para medir vibraciones en sus equipos.

⁹ Diagrama interno de un ejemplo real de acelerómetro capacitivo del fabricante PCB Piezotronics. (a) Acelerómetro en reposo. b) Acelerómetro sometido a una aceleración en sentido ascendente.



Cap.2: Fig. 21 Esquema interno básico de un sensor tipo MEMS.

FUENTE: (37)

En Fig.21, la barra voladiza sostiene la micro-masa inercial entre dos electrodos capacitivos que permiten convertir el movimiento en una señal eléctrica.

TIPO DE ACELERÓMETRO	MARGEN DE MEDIDA	ANCHO DE BANDA (HZ)	VENTAJAS Y DESVENTAJAS	APLICACIONES
Micromecánico MEMS	De 1,5 a 250g	De 0,1 a 1500	- Alta sensibilidad - Costo medio - Uso sencillo - Bajas temperaturas	- Impacto - ABS - Airbag - Automoción
Piezo-eléctricos	De 0 a 2000g	De 10 a 20000	- Sensibilidad media - Uso complejo - Bajas temperaturas - No funcionan en DC	- Vibración - Impacto - Uso industrial
Piezo-resistivos	De 0 a 2000g	De 0 a 10000	- Respuesta en DC y CA - Prestaciones medias - Bajo costo	- Vibración - Impacto - Automoción
Capacitivos	De 0 a 1000g	De 0 a 2000	- Funciona en DC - Bajo ruido - Baja potencia - Excelentes características	- Uso general - Uso industrial
Mecánicos	De 0 a 200g	De 0 a 1000	- Alta precisión en DC - Lentos - Alto costo	- Navegación inercial - Guía de misiles - Herramientas - Nivelación

Cap.2: Tabla. 3 Principales características y aplicaciones de los acelerómetros considerando el margen de medida g.

FUENTE: (38)

Luego de revisar los diferentes tipos de acelerómetros: según la forma de medida (dinámica y estática), y según el número de ejes que pueden sensar (monoaxial, biaxial y triaxial); se obtiene un resumen de los diferentes tipos de acelerómetros mostrados en la Tabla 3. En base a este análisis y de acuerdo a las características e indicadores del proyecto, se determina la conveniencia de utilizar el acelerómetro tipo MEMS.

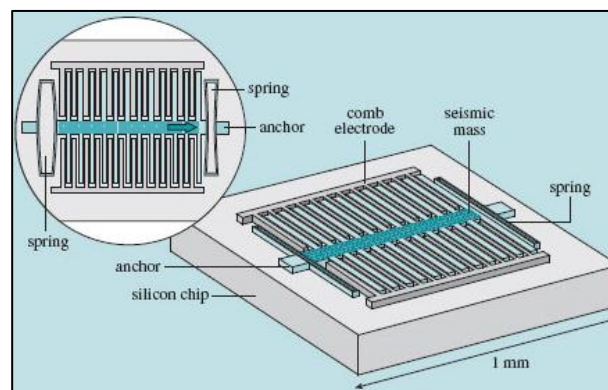
2.5.1.5 ACELERÓMETRO MEMS

Se encuentran entre los primeros productos de microsistemas (MEMS). Miden el movimiento, la aceleración, vibración, choque e inclinación. Con relación a esta tecnología, se distingue dos categorías principales de acelerómetros MEMS: el capacitivo de silicio y el piezorresistivo, siendo el primero el que domina actualmente el mercado.

Un acelerómetro MEMS consiste en un pequeño chip que está enteramente hecho de silicio y consiste en dos partes fundamentales que son:

La primera es una masa (frecuentemente denominada masa de prueba o masa sísmica) suspendida por medio de un resorte en cada extremo.

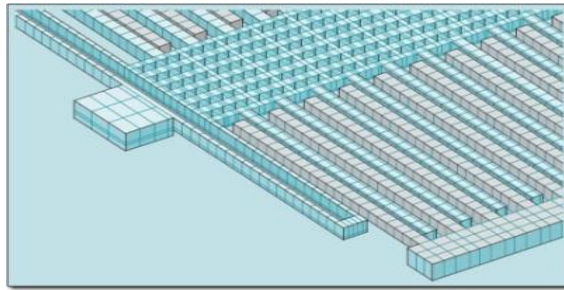
La segunda parte es un par de sensores denominados electrodos fijos que permiten la electrónica para detectar el movimiento relativo de la masa con respecto a la plataforma de silicio.



Cap.2: Fig. 22 Detalle de un típico acelerómetro MEMS.

FUENTE: (39).

Cuando el chip es sometido a una aceleración, la masa de prueba (Fig. 23) se mueve un poco con relación al chip y las estructuras fijas en él. La cantidad de movimiento depende del tamaño de la aceleración, la rigidez del resorte y la masa. Cuando la masa se desvía, la capacitancia eléctrica entre ésta y las estructuras de sentido en el chip cambia, y este cambio es detectado mediante un acondicionamiento electrónico que lo convierte en un valor representativo de la aceleración.



Cap.2: Fig. 23 Masa sísmica de un acelerómetro MEMS.

FUENTE: (39).

2.5.1.5.1 CARACTERÍSTICAS DE LOS ACELERÓMETROS MEMS

Dos parámetros principales a la hora de escoger el sensor adecuado son los rangos de funcionamiento de temperatura y frecuencia, tomando en cuenta también el tamaño, gravedad, resistencia a golpes y el precio. La correcta elección depende de la eficiencia y utilidad que se obtiene como resultado de usar un determinado acelerómetro.

▪ SENSIBILIDAD

La sensibilidad de los acelerómetros permite saber la cantidad de medida que percibe el sensor en función de la magnitud física que se aplica sobre el dispositivo. Esta cantidad de medida es el valor de voltaje que el acelerómetro proporciona por cada unidad de gravedad aplicada (mV/g). Donde mV (unidad eléctrica que se utiliza) con respecto a la magnitud física que en este caso la aceleración en g ($1g = 9.81m/s^2$).

En los acelerómetros digitales como en la IMU MPU-6050 dispositivo usado en este trabajo la sensibilidad se mide en LSB/mg .

Generalmente el valor de la sensibilidad se asocia con un intervalo de frecuencia y amplitud. De esta manera en un mismo acelerómetro la sensibilidad suele ser distinta, dependiendo de las características del movimiento y de parámetros ambientales como la temperatura; por ejemplo: $100mV/g$ a una frecuencia de 100Hz para un nivel de aceleración pico de: $1g$ a $24^{\circ}C$.

▪ **LSB (Least Significant Bit)**

Con lo que concierne a los ADC, el LSB es la variación de voltaje que produce un cambio en una unidad en la salida del ADC, o también se determina, como el rango de voltaje designado a cada bit. Su valor depende del rango máximo de voltaje y del número de bits del ADC.

▪ **RESOLUCIÓN**

Nivel más bajo de gravedad que el acelerómetro es capaz de medir. Es decir el cambio más pequeño en la entrada que es detectada a la salida, esta relacionada con el voltaje sobre la unidad de gravedad V/g .

Para sensores puramente analógicos este parámetro es infinito, mientras que en aquellos que implementan circuitos digitales con sus respectivos conversores ADC la resolución está en función del número de bits utilizados para el procesamiento.

$$Resolución = 1LSB = V/2^n \text{ bits}$$

▪ **ANCHO DE BANDA**

Es el rango de frecuencia máxima que mide un dispositivo, indicando cómo este responde a frecuencias diferentes. A mayor ancho de banda los sensores miden la frecuencia más alta de movimiento y vibración, siendo el ruido eléctrico un inconveniente, el cual se reduce o elimina usando un filtro paso bajo.

▪ **RANGO DE MEDIDA**

Se refiere a la cantidad de gravedades que el dispositivo cuenta y la medida máxima que soporta el sensor. Para los acelerómetros el rango de medición se evalúa con la gravedad estándar $g = 9.80 \text{ m/s}^2$. Existen acelerómetros de 1,5g, 2g, 4g, etc. El nivel de rango de medida determina la sensibilidad, a mayor rango, menor sensibilidad.

▪ **DENSIDAD DE RUIDO**

Este parámetro influye en la calidad de la señal, este valor está relacionado con el ancho de banda elegido.

▪ **CALIBRACIÓN**

La calibración debe efectuarse de manera sencilla y el sensor no debe precisar una recalibración frecuente.

▪ **FIABILIDAD**

El sensor no debe estar sujeto a fallos inesperados durante su funcionamiento.

2.5.1.5.2 CALIBRACIÓN

La calibración es el proceso de comparar la salida del instrumento con una información de referencia conocida y determinar los coeficientes que fuercen su salida. En los acelerómetros se extrae el coeficiente de transferencia que relaciona la señal de salida del transductor con la aceleración a la que es sometido. Este coeficiente de transferencia conocido como sensibilidad se conoce de diversas maneras; esto depende del tipo de acelerómetro, de su aplicación y hasta del método de calibración. Los parámetros que generalmente se consideran para la calibración de los acelerómetros son: el rango de frecuencia, los niveles de aceleración, la resolución y la precisión.

A) MÉTODO DE CALIBRACIÓN

▪ **CALIBRACIÓN ABSOLUTA**

Mide la proyección de la aceleración gravitacional mediante un patrón de medida trazable, es decir la sensibilidad del acelerómetro se determina a partir de mediciones basadas en unidades fundamentales y derivada de las propiedades físicas involucradas. Consiste en someter al acelerómetro a una vibración senoidal mientras se registra el desplazamiento en el tiempo y así extraer la sensibilidad del acelerómetro para verificar su respuesta en amplitud y frecuencia.

▪ **CALIBRACIÓN POR COMPARACIÓN (BACK TO BACK)**

La sensibilidad se mide a partir de la relación que se establece con un acelerómetro cuyos parámetros son previamente calibrados, con una calidad igual o superior en lugar de usar patrones directamente. Este método simplifica las operaciones y reduce costos.

▪ **CALIBRACIÓN POR RECIPROCIDAD**

Consiste en calibrar previamente el aparato que genera la señal de referencia, de manera que esta señal se relacione con la aceleración de referencia que entra en el sensor.

▪ **CALIBRACIÓN POR INCLINACIÓN PARA MEDIDAS ESTÁTICAS**

Es el método más tradicional y simple aplicado para comprobación preliminar y rápida. Se coloca el acelerómetro sobre una mesa inclinable en dos ejes. El plano de la mesa sirve como soporte para un giro completo del acelerómetro. El principal inconveniente es que solo produce señales estáticas, ya que no se logra analizar su respuesta con respecto a la frecuencia.

Generalmente el comportamiento del acelerómetro se obtiene según la frecuencia de la señal de entrada, la razón es que, las aplicaciones en las que se usa el acelerómetro (sus aceleraciones) varían rápidamente en el tiempo.

B) ERRORES DE CALIBRACIÓN

▪ **SESGO (BIAS)**

El sesgo es la medida sobre un tiempo determinado de la salida del sensor en condiciones específicas de funcionamiento que no tienen correlación con la entrada, esta medida se toma en el proceso de calibración del sensor y luego es corregido en

el procesamiento del INS. Es definido también como un error sistemático¹⁰ del cero del instrumento por causas de funcionamiento interno y resultante de la sensibilidad a campos externos, El sesgo se lo expresa como: m/s^2 o 1g. Cuando el sesgo es constante se denomina *offset*; esta es una medida proporcionada por el sensor y se la determina por calibración o la hoja de datos del fabricante (*data sheet*). Mediante la siguiente ecuación se determina cuando el sesgo es constante o no; esto se comprueba mediante la simetría de los sesgos positivo y negativo paralelos a la gravedad. Siendo g^+ y g^- los valores medios del sensor en paralelo a la dirección y en contra del vector gravedad.

$$sesgo (b) = \frac{g^+ + g^-}{2}$$

▪ FACTOR DE ESCALA

Denominado también *sensibilidad*, es la razón de cambio en la salida con respecto al cambio en la entrada del sensor objeto de medición. Se trata de una constante multiplicativa que escala el valor real de la medición del sensor respecto a la medición esperada teóricamente. El factor de escala se ve afectado por factores relacionados con el material y su construcción.

$$escala (s) = \frac{g^+ + g^-}{2g}$$

▪ DERIVA (DRIFT)

Se presenta debido a la acumulación de errores sistemáticos (*bias*) que van siendo integrados en el tiempo, es decir el error se va incrementando mientras transcurre el tiempo.

¹⁰ Un error es sistemático cuando en el curso de varias mediciones de una magnitud de un determinado valor, realizadas en las mismas condiciones; permanece constante con valor absoluto y signo o varía de acuerdo de como cambian las condiciones de medida. El resultado de la medida influye tanto en el aparato y método empleado para efectuar la media, así también como el operario y toda una serie de circunstancias (climáticas, mecánicas, electrónicas, etc.).

▪ DESALINEACIÓN

La desalineación de los ejes es el error resultante de la imperfección del montaje de los sensores; presentando una no ortogonalidad en el rango de operación del sensor, siendo afectada la medida de cada eje, por la medida de los otros dos ejes del sensor. Existe un error de desalineación especificado por el fabricante que es del 5%, esto quiere decir que si el sensor del eje Z esta paralelo a la gravedad, los sensores f_x y f_y de los ejes X y Y, medirán una aceleración de 0.05g que equivale a una pérdida de ortogonalidad de 2.87 grados en los ejes X y Y.

▪ RUIDO TÉRMICO

Este ruido es generado por energía térmica debido a la agitación del movimiento aleatorio de transporte de carga dentro de los acelerómetros. El ruido térmico esta representado por la siguiente ecuación, que depende de la frecuencia de Nyquist que esta relacionada con el ancho de banda de la señal (B).

$$N = \frac{350 \mu g}{\sqrt{B(Hz)}}$$

2.5.2 EL GIROSCOPIO

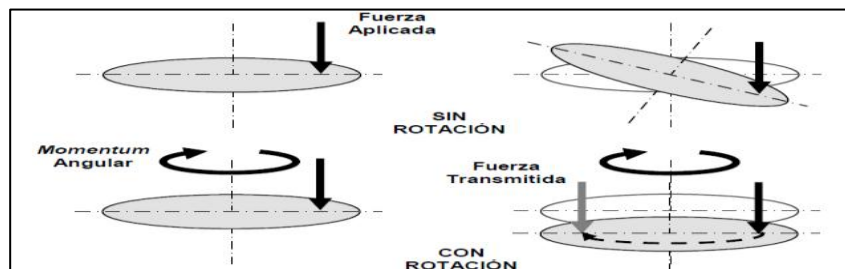
El giroscopio se constituye en uno de los dispositivos importantes en lo que concierne a navegación inercial, debido a la cualidad que tiene de mantener la dirección de rotación a través de un eje que atraviesa su centro de masa aun cuando el marco referencial rote. El nombre de "giróscopo" proviene del griego mediante la combinación de las palabras " skopein ", que significa ver y "gyro" que significa giro que fue inventado por el astrónomo León Foucault en 1852 durante sus experimentos para medir la rotación de la Tierra.

El giroscopio más sencillo consiste en una masa girando a alta velocidad contenida en un sistema mecánico que le permite total libertad de rotación.

El giroscopio es un dispositivo que permite conocer como varía un ángulo en el tiempo mientras se encuentra rotando, basado en los principios de conservación del

momento angular¹¹. Cuando se somete el giroscopio a un momento de fuerza que tiende a cambiar de dirección como lo haría un cuerpo que no girase, cambia la orientación en una dirección perpendicular a la dirección intuitiva.

La conservación del momento angular¹² enuncia que si la torsión externa neta aplicada a un sistema es cero, el momento angular interno no cambiará, esto engloba tanto la dirección como la magnitud de dicho momento.



Cap.2: Fig. 24 Funcionamiento de un giroscopio.

FUENTE: (40).

En Fig. 24, se observa que si la masa se encuentra estática, al aplicar una fuerza en uno de sus extremos (fuerza gravitacional), este se inclinará, sin embargo si la masa posee un momentum angular, debido a la rotación misma la fuerza será transmitida por todo el disco, siguiendo una circunferencia cuyo radio esta definido por el punto en donde se ha aplicado la fuerza. Este fenómeno es equivalente a que si la fuerza se aplicaría por igual a todos los puntos de la circunferencia trazada, de esta manera el giroscopio mantiene su dirección de rotación

2.5.2.1 CLASIFICACIÓN

Hay algunas clases de giroscopios que operan siguiendo diferentes principios pero en general son agrupados en tres principales categorías como son los giroscopios mecánicos ópticos y vibratorios.

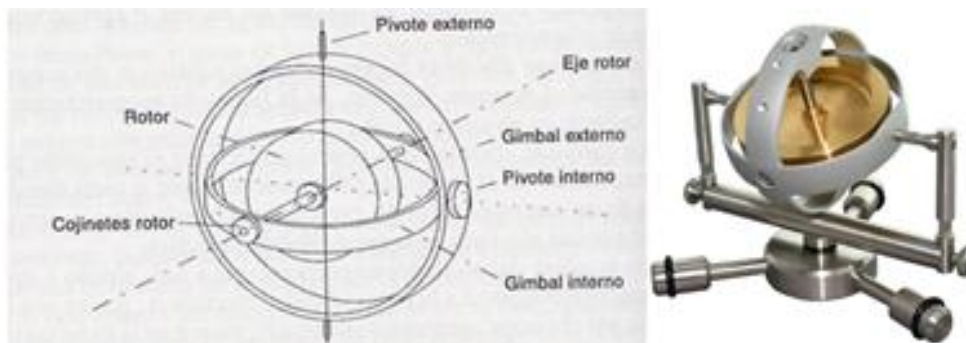
¹¹ Se dice momento angular de un cuerpo que gira, por ejemplo una estrella que gira alrededor de sí misma, al producto de la masa m por el radio r , por su velocidad de rotación v .

¹² La conservación del momento angular dice, que si un cuerpo que gira se contrae, es decir, si la masa que lo forma se reúne en el centro, la velocidad de rotación aumenta de manera que el momento angular resultante se mantiene inalterado, y a la inversa, si la masa se distribuye hacia la periferia, la velocidad de rotación disminuye de manera que el momento angular se mantiene.

▪ GIROSCOPIO ROTATORIOS MECÁNICOS

También conocido como giroscopio mecánico de Foucault, se basa en una masa rotando sobre un eje sostenida por varios cardanes (gimbals) o sistema mecánico que le permite total libertad de rotación.

Cuando la rueda esta girando, ésta experimenta un momento angular grande que permite mantener su orientación casi fija cuando un torque externo se aplica a su estructura. Desafortunadamente los giroscopios mecánicos son demasiado grandes y son imprácticos para usar en pequeños dispositivos como teléfonos celulares o mouse de computadora, etc.



Cap.2: Fig. 25 Giroscopio mecánico.

FUENTE: (41).

▪ GIROSCOPIO ÓPTICO

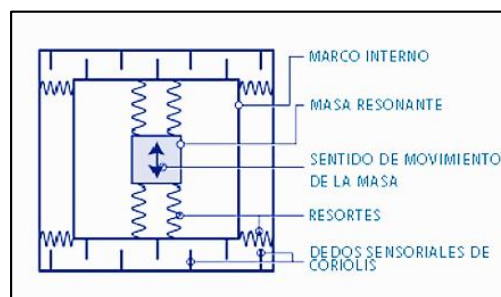
Usa el efecto Sagnac para detectar la rotación a la cual están siendo sometidos. El efecto Sagnac se produce cuando dos rayos de luz circulan en direcciones opuestas dentro de un camino cerrado, el haz de luz que circula en la misma dirección de la rotación tarda más tiempo en viajar que el haz de luz que va en sentido contrario, con lo cual el ángulo se obtiene en base a la diferencia de camino que ven los dos rayos que viajan en direcciones opuestas a lo largo del perímetro.

Al eliminar virtualmente todas las limitaciones mecánicas tales como las vibraciones, sensibilidad al choque y la fricción, los giroscopios ópticos se encuentran en muchas aplicaciones de gama alta a pesar de sus costos elevados.

▪ GIROSCOPIO VIBRATORIO

En los últimos años se ha introducido la estructura de los giróscopos vibratorios, que son fabricados usando tecnología MEMS.

Los giróscopos vibratorios se caracterizan por disponer de un elemento vibrante que al ser forzado a rotar son afectados por una fuerza de Coriolis, que induce vibraciones secundarias ortogonales a la vibración original. La vibración angular se obtiene en base de estas vibraciones.



Cap.2: Fig. 26 Esquemático de la estructura vibratoria de un giroscopio MEMS

FUENTE: (42).

2.5.2.2 ANÁLISIS DE TIPOS DE GIROSCOPIOS

TIPO DE GIROSCOPIO	CARACTERÍSTICAS
MECÁNICO	<ul style="list-style-type: none"> ▪ Poseen partes móviles ▪ Lectura analógica
ÓPTICO	<ul style="list-style-type: none"> ▪ Gran rango dinámico ▪ Lectura digital ▪ bajo costo ▪ tamaño reducido ▪ No es necesario un tiempo de calentamiento
VIBRATORIO (MEMS)	<ul style="list-style-type: none"> ▪ tamaño reducido ▪ bajo costo ▪ Sin piezas móviles ▪ Baja potencia

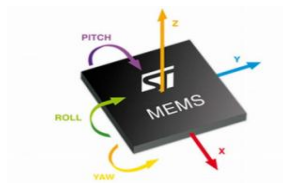
Cap.2: Tabla. 4 Tabla comparativa de tipos de giroscopio.

FUENTE: AUTOR.

Luego de analizar las características más importantes de los diferentes tipos de giróscopos; se ha decidido usar el giroscopio tipo MEMS, detallado a continuación.

2.5.2.3 GIROSCOPIO MEMS

Es un circuito integrado de detección de inercia que mide el ángulo y el índice de rotación en un objeto o sistema. El giroscopio MEMS tipo vibrante piezoeléctrico mide la velocidad angular (Pitch, Roll y Yaw) usando el efecto Coriolis.



Cap.2: Fig. 27 Velocidades angulares que mide un giroscopio.

FUENTE: (40).

2.5.2.3.1 EFECTO CORIOLIS

El efecto Coriolis, descrito en 1836 por el científico francés Gaspard-Gustave Coriolis, es una fuerza ficticia que hace que un objeto que se mueve sobre el radio de un disco en rotación tienda a acelerarse con respecto a ese disco dependiendo si el movimiento es hacia el eje de giro o si está alejándose de este.

▪ FUERZA DE CORIOLIS

Es la aceleración que sufre un objeto desde el punto de vista del observador en rotación. Es como si para el observador existiera una fuerza sobre el objeto que lo acelera. Es una fuerza ficticia que aparece cuando un cuerpo esta en movimiento con respecto a un sistema en rotación y se describe su movimiento en esa rotación.

$$F_c = -2m(w \times v)$$

m , masa del cuerpo.

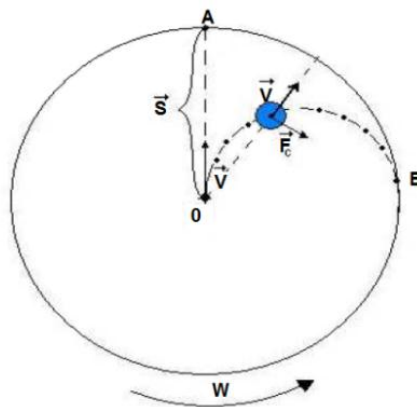
v , velocidad del cuerpo en el sistema de rotación.

w , velocidad angular del sistema en rotación vista desde un sistema inercial.

\times , producto vectorial.

La fuerza de Coriolis es diferente de la fuerza centrífuga, porque es siempre perpendicular a la dirección del eje de rotación del sistema y a la dirección del movimiento del cuerpo vista desde el sistema en rotación.

En la siguiente figura se observa un experimento en donde un disco gira con una velocidad angular ω , respecto a un eje perpendicular a la superficie del disco. En la superficie del disco se encuentra una bola de masa m , que se desplaza a una velocidad v , en la dirección que se observa en la figura, en ésta; se observa la trayectoria seguida por la bola en la superficie del disco debido al efecto de las fuerzas de Coriolis (43).



Cap.2: Fig. 28 Fuerzas que actúan en el efecto Coriolis.

FUENTE: (43).

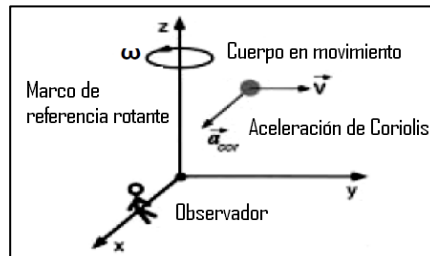
En Fig.28, se demuestra que sin el efecto de las fuerzas de Coriolis la bola partiría de O hasta A, debido a estas fuerzas la bola acabaría en B.

La fuerza de Coriolis tiene dos componentes:

- Una componente tangencial, debida a la componente radial del cuerpo.
- Una componente radial, debida a la componente tangencial del movimiento del cuerpo.

▪ ACELERACIÓN CORIOLIS

Todo cuerpo que se mueve sobre un sistema en rotación experimenta una aceleración, llamada aceleración Coriolis.



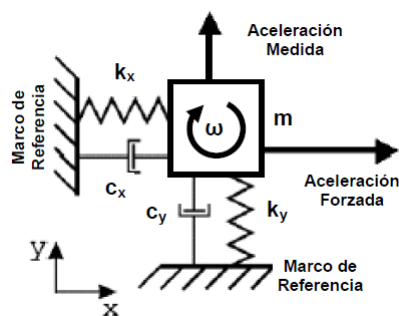
Cap.2: Fig. 29 Aceleración Coriolis con respecto a un observador.

FUENTE: (40).

En Fig.29, se asume que existe una partícula que se mueve con una velocidad v , medida con respecto a un marco de referencia que es afectado por una velocidad angular w , en este marco existe un observador, que percibe a la partícula como si experimentara un cambio en su dirección debido a una aceleración que se expresa mediante la fórmula

$$\bar{a}_{cor} = 2\bar{v} \cdot \bar{w}$$

Si se consideran conocidos los valores de la velocidad de dicha partícula y de la aceleración de Coriolis, se podría hallar fácilmente la velocidad angular y aplicando una integral en función del tiempo, se obtendría el desplazamiento angular. El modelo que permite determinar la velocidad angular, a través del cálculo de la aceleración de Coriolis se ilustra en Fig.30.



Cap.2: Fig. 30 Modelo simplificado para un giróscopo de estructura vibratoria.

FUENTE: (40).

En Fig.30, sugiere un sistema oscilante de segundo orden con dos grados de libertad, a uno de los cuales se le ha aplicado una aceleración de tipo sinusoidal,

por lo tanto en dicha coordenada se obtiene un sistema oscilante forzado. Las ecuaciones simplificadas que gobiernan el sistema se describen a continuación:

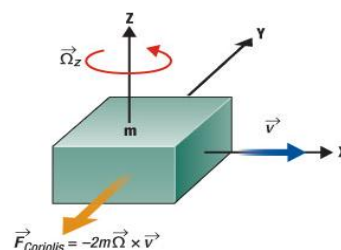
$$m\ddot{x} + c_x\dot{x} + k_x x = \tau_x + 2m\dot{y}\omega$$

$$m\ddot{y} + c_y\dot{y} + k_y y = \tau_y + 2m\dot{x}\omega$$

En el lado derecho de cada una de las ecuaciones, el segundo miembro, representa la aceleración inducida por el efecto de Coriolis, que produce un acoplamiento dinámico entre los ejes de oscilación. Para un mejor entendimiento de todo esto, se asume que la masa está oscilando a lo largo del eje X , en este momento se aplica una velocidad angular constante al marco de referencia, en consecuencia de esto el sistema comienza a oscilar a lo largo del eje Y . La aceleración producida por esta oscilación se calcula y es proporcional a la velocidad angular del marco de referencia. Claramente se nota que la masa se opone al cambio de la dirección de oscilación.

2.5.2.3.2 EFECTO CORIOLIS EN LOS MEMS

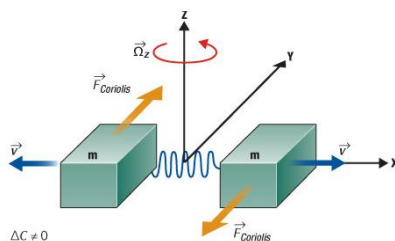
Para deducir cómo se produce el efecto Coriolis en el interior del MEMS se considera el siguiente bloque de masa m , el cual se mueve a una velocidad v , como se observa en la figura a continuación presentada. Si a este bloque se aplica un movimiento angular Ω_z , se producirá una fuerza de Coriolis de valor $F = -2mV \times W$, en la dirección que apunta F .



Cap.2: Fig. 31 Efecto Coriolis.

FUENTE: (44).

En la práctica, son dos masas las que utilizan los MEMS, como se observar en Fig.32.

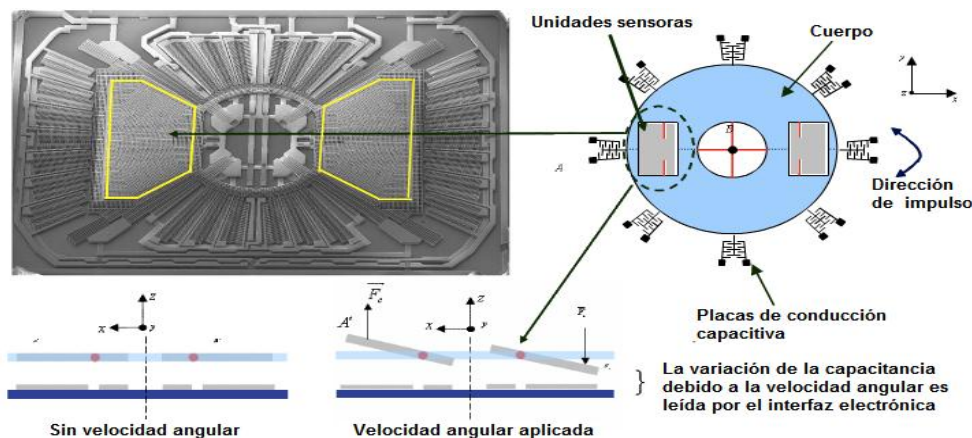


Cap.2: Fig. 32 Velocidad angular aplicada.

FUENTE: (44).

Las dos masas oscilan de forma constante a una velocidad v , cuando se le aplica una velocidad angular, que viene denotada por Ω_z , las fuerzas de Coriolis resultantes tienen sentido opuesto. Este resultado se traduce a una medida diferencial capacitiva, Cuando una aceleración es aplicada a las dos masas en la misma dirección, estas se mueven en la misma dirección, por lo que la medida capacitiva diferencial es cero

En resumen en los giroscopios MEMS las masas se encuentran continuamente en movimiento, cuando un movimiento angular es aplicado, se genera un par de fuerzas de Coriolis, las cuales son medidas mediante una interfaz sensora capacitiva. Estas fuerzas son proporcionales a la velocidad angular aplicada, que es la magnitud que interesa medir.¹³



Cap.2: Fig. 33 Funcionamiento interno del giroscopio MEMS.

FUENTE: (45).

¹³ Lo que el giroscopio entrega es la medida de la velocidad angular, si se integra ese valor, se obtiene el ángulo de navegación.

Cuando se aplica una velocidad angular se produce el par de fuerzas de Coriolis que desplazan las placas tal como se observa en Fig.33. Este desplazamiento produce una variación en la distancia entre las placas del condensador, lo que implica una variación de la capacidad del condensador.

Las unidades en que se mide la variación del giro son (rad/s). El fabricante presenta sus especificaciones referidas a grados, pero para procesar los datos se suele trabajar en radianes.

2.5.2.4 APLICACIONES

Los giróscopos son partes muy importantes de los sistemas de navegación de barcos, guías inerciales de aviones, naves espaciales, submarinos, misiles teledirigidos. Navegación (INS) cuando brújulas magnéticas no funcionan (como en el telescopio Hubble) o no son suficientemente precisos (como en misiles balísticos intercontinentales) o para la estabilización de vehículos voladores, como los helicópteros de radio control o vehículos aéreos no tripulados. Debido a una mayor precisión, los giroscopios también se utilizan para mantener la dirección en la minería del túnel.

2.6 REGISTRO DE LA ORIENTACIÓN

El registro de la orientación mediante el acelerómetro y el giroscopio para sensar la posición de un objeto en el espacio tridimensional es efectuado por el cálculo del ángulo de inclinación (actitud).

2.6.1 EL ACELERÓMETRO Y LA INCLINACIÓN

2.6.1.1 CÁLCULO DE LA INCLINACIÓN USANDO SOLAMENTE UN ACELERÓMETRO

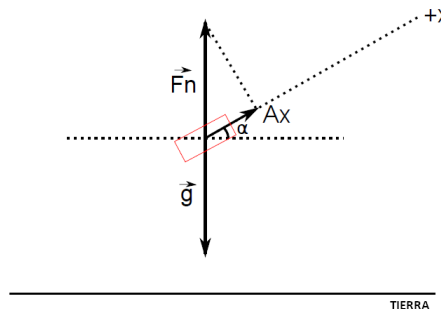
Las salidas de un acelerómetro de tres ejes están sujetas a la fuerza de gravedad g , por lo tanto es utilizada para la detección de la inclinación, de esta manera se

calcula Pitch, definido como el ángulo entre el eje X y el plano horizontal, y Roll, definido como el ángulo entre el eje Y y el plano horizontal.

A. DETECCIÓN DE LA INCLINACIÓN BASADO EN UN SOLO EJE

Usado para aplicaciones donde es necesaria la detección de inclinación sobre un ángulo limitado mediante la utilización de un solo eje del acelerómetro, En Fig.34, la fuerza de la gravedad g es contrastada mediante la fuerza normal F_n , de manera que $g = F_n$. En realidad, el acelerómetro mide la fuerza normal que actúa sobre la superficie en la que se encuentra el dispositivo **(46)**.

Cuando se inclina el acelerómetro un ángulo α , la salida del acelerómetro A_x será la proyección de la fuerza normal F_n sobre el eje X del acelerómetro.



Cap.2: Fig. 34 Cálculo de la inclinación usando un sólo eje del acelerómetro.

FUENTE: (46).

Para un valor imaginario de gravedad de $1g$, se deduce fácilmente la aceleración de salida mediante la siguiente relación:

$$A_x[g] = 1g * \sin(\alpha)$$

Por lo tanto, el ángulo de inclinación se calcula mediante:

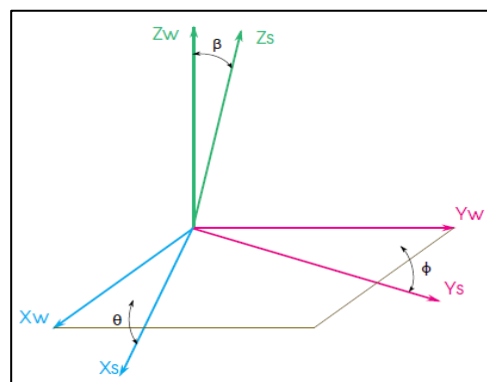
$$\alpha = \arcsin\left(\frac{A_x[g]}{1g}\right)$$

Debido a que este método utiliza solamente un eje y requiere del vector gravedad, el cálculo del ángulo de inclinación es preciso solamente cuando el sensor está

orientado de manera que el eje X siempre permanezca en el plano de la gravedad. Cualquier rotación alrededor de los otros ejes reduce la magnitud de la aceleración en el eje X , resultando un error en los cálculos del ángulo de inclinación. Con el fin de eliminar esta restricción es necesario realizar los cálculos en los tres ejes del sensor.

B. DETECCIÓN DE LA INCLINACIÓN BASADO EN TRES EJES.

Cuando se inclina un acelerómetro triaxial, la fuerza normal F_n se proyectará sobre todos los ejes del dispositivo. El problema para determinar pitch (θ), ángulo entre el eje X y el plano horizontal, y roll (ϕ) ángulo entre el eje Y y el plano horizontal, se resuelve geoméricamente (46).



Cap.2: Fig. 35 Cálculo de la inclinación empleando un acelerómetro de 3 ejes.

FUENTE: (46).

Las ecuaciones que permiten hallar los ángulos de inclinación con respecto a cada uno de los planos se describen a continuación:

$$\text{pitch} = \theta = \arctan\left(\frac{A_x}{\sqrt{A_y^2 + A_z^2}}\right)$$

$$\text{roll} = \phi = \arctan\left(\frac{A_y}{\sqrt{A_x^2 + A_z^2}}\right)$$

$$\beta = \arctan\left(\frac{\sqrt{A_x^2 + A_y^2}}{A_z}\right)$$

2.6.1.2 LIMITACIONES AL USAR SOLAMENTE UN ACELERÓMETRO PARA EL CÁLCULO DE LA INCLINACIÓN.

- El censado de la inclinación con un acelerómetro es simple y directo; posible solo en ausencia de movimiento, es decir, sin la presencia de una aceleración lineal en cualquiera de sus ejes. Suponiendo que el dispositivo está siendo utilizado en un entorno con vibraciones (por ejemplo en un carro o un avión), la presencia tan solo de dichas vibraciones producirá que las medidas obtenidas de los ángulos sean poco fiables, por lo tanto, la información seguirá siendo errónea mientras el dispositivo se encuentre bajo estos efectos. El promedio de varias lecturas del acelerómetro ayuda a filtrar algunas aceleraciones externas, pero en general las fórmulas presentadas anteriormente son válidas solamente si se asume que el dispositivo esta casi estable.
- Otra restricción de usar solo un acelerómetro es que no es posible obtener información sobre yaw definido como el ángulo de rotación con respecto al eje vertical.

2.6.2 EL GIROSCOPIO Y LA INCLINACIÓN

2.6.2.1 CÁLCULO DE LA INCLINACIÓN USANDO SOLAMENTE UN GIRÓSCOPO

Para obtener el ángulo de inclinación es necesario integrar la velocidad angular, tomando en cuenta que la definición de integral implica multiplicar un valor por un tiempo dt . El valor constante de tiempo depende de la frecuencia de muestreo a la que está ejecutando la aplicación.

$$\frac{d\theta}{dt} = \text{velocidad angular} = \text{salida del giróscopo}$$

$$d\theta = \text{velocidad angular} * dt$$

Dado que la integral se define como la inversa de la derivada, la relación anterior queda como sigue:

$$\int \text{velocidad angular} = \int \text{salida del gir6scopo} = \theta$$

El proceso para encontrar el 6ngulo de inclinaci6n es c6clico, se lo efectúa a intervalos de tiempo establecidos dt mediante la implementaci6n de un algoritmo acumulativo siguiendo el siguiente modelo matem6tico.

$$\theta_k = \theta_{k-1} + u_k * dt$$

Donde:

u_k = seál entregada por el gir6scopo.

dt = tiempo de muestreo.

θ_{k-1} = medida del 6ngulo previo.

θ_k = medida del 6ngulo actual.

2.6.2.2 LIMITACIONES AL USAR SOLAMENTE UN GIROSCOPIO PARA EL CÁLCULO DE LA INCLINACIÓN

- La integraci6n de las medidas de un gir6scopo dan lugar a la acumulaci6n del error de deriva en la orientaci6n calculada. El algoritmo produce una suma continua de las lecturas del sensor y con ellas tambi6n el error de deriva, estos valores ser6n v6lidos 6nicamente para pequeos intervalos de tiempo, mientras que para intervalos m6s largos de tiempo estos valores estar6n totalmente desfasados con respecto a la medida real.
- Otro inconveniente que se presenta al usar solamente un gir6scopo es que no se logra establecer un estado de referencia inicial o punto de partida, se corrige este problema mediante el uso de un aceler6metro o un magnet6metro. Por lo tanto los gir6scopos por si solos no proporcionan una medida absoluta de la orientaci6n **(46)**.

2.6.3 CÁLCULO DE LA INCLINACIÓN USANDO GIRÓSCOPO Y ACELERÓMETRO

Al analizar los dos métodos anteriores se determina claramente que mediante el uso tanto del acelerómetro y giróscopo de forma independiente, se presentan varias restricciones que hacen que el resultado final produzca errores en su magnitud.

La solución más viable para corregir estos inconvenientes es trabajar con los dos sensores simultáneamente, sin embargo es probable que al estar sujetos a altos niveles de ruido, por ejemplo: las aceleraciones debido al movimiento corromperán la medida de la dirección y de la gravedad, por tanto surge la necesidad del emplear un “filtro de orientación” que consiste en calcular una simple estimación de la orientación a través de una fusión óptima de las lecturas del giróscopo y el acelerómetro **(47)**.

Hasta esta fase el filtro de Kalman se avisa como la mejor opción para fusionar las lecturas de ambos sensores.

2.6.3.1 FILTRO DE KALMAN

El filtro de Kalman se ha convertido en la base aceptada para la mayoría de algoritmos de filtros de orientación y sensores comerciales de orientación inercial como Xsens, VectorNav, Intersense, PNI y Crossbow.

El uso extendido de soluciones basadas en Kalman son testimonios de su exactitud y eficacia, sin embargo tienen una serie de desventajas. Resulta ser muy complicado para su implementación. Las iteraciones de regresión lineal, fundamentales para el proceso de Kalman, demandan frecuencias de muestreo muy altas, es sensible al fenómeno conocido como gimbal lock (bloqueo del cardán), requiere configurar varios parámetros, todos ellos matrices, por lo tanto, sus cálculos exigen gran carga computacional.

Por todas las desventajas descritas, el filtro de Kalman no es una solución óptima para ser utilizada en el proyecto, surgiendo así la motivación de buscar enfoques alternativos.

2.6.4 FILTRO BASADO EN GRADIENTE DESCENDENTE

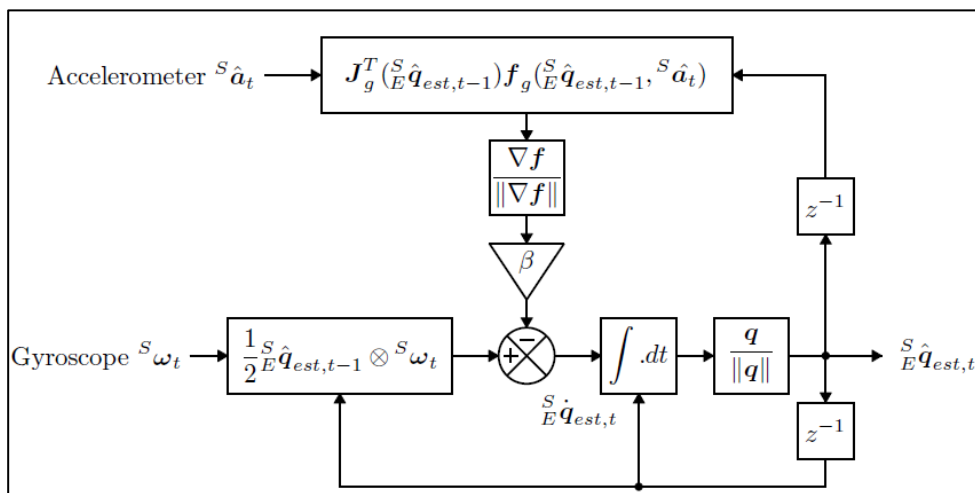
En el año 2009 Sebastian Madgwick en su publicación (An efficient orientation filter for inertial and inertial/magnetic sensor arrays) desarrolla un algoritmo de filtro aplicable a IMUs, sensor formado por un giróscopo y un acelerómetro y a MARGs (Magnetic, Angular Rate and Gravity) una IMU híbrida que incorpora también un magnetómetro, como parte de su investigación de doctorado en la Universidad de Bristol **(46)**.

El algoritmo de filtro propuesto por Madgwick en su publicación, presenta 4 ventajas importantes que claramente es la mejor opción para ser utilizado en el desarrollo del proyecto:

- Uso de Cuaterniones.
- Carga computacional mínima, el algoritmo solo requiere de 109 operaciones aritméticas en cada iteración.
- Requiere de un solo parámetro para configurar el filtro.
- El algoritmo operar en muestreos de baja frecuencia; ejemplo 10HZ.

2.6.4.1 ALGORITMO DE FUSIÓN

Existen muchos algoritmos de optimización pero el algoritmo de gradiente descendente es uno de los más simples en la implementación y cálculo. A diferencia del filtro Kalman, este algoritmo no es un predictor, únicamente fusiona las dos entradas tanto las del giroscopio como las del acelerómetro empleando técnicas como gradiente descendente o matriz Jacobiana (una matriz formada por derivadas parciales de primer orden en una función). El diagrama de bloques de la siguiente figura ilustra el proceso de funcionamiento del algoritmo.



Cap.2: Fig. 36 Implementación de un filtro de orientación para una IMU (Giróscopo + Acelerómetro).

FUENTE: (46).

2.7 PROTOCOLOS DE COMUNICACIÓN

El protocolo de comunicación es un conjunto de reglas, necesarias para establecer la comunicación entre dispositivos, que permite un intercambio de datos fiables.

2.7.1 PROTOCOLO DE COMUNICACIÓN RS232

La norma RS-232 fue originalmente pensada para regir las comunicaciones entre ordenadores y equipos de módem de la época (hace más de 40 años).

El RS232 es un protocolo de comunicación serial orientado a caracteres, es decir, un protocolo donde toda la información es enviada por un solo canal bit a bit (un canal para enviar información y otro para recibirla), y donde lo que se envían son caracteres. Por ejemplo, si se envía el número 345, primero se debe que enviar el carácter 3, seguidamente el 4 y para finalizar el 5, y no el byte que represente el número 345. RS-232 está diseñado para distancias cortas, de unos 15 metros aproximadamente, trabaja de forma asíncrona o síncrona y con tipos de canal simplex, halfduplex y fullduplex.

2.7.1.1 HARDWARE DEL PROTOCOLO RS-232

Una conexión RS232 está definida por un cable desde un dispositivo al otro. Hay 25 conexiones en la especificación completa pero en la mayoría de los casos se utilizan menos de la mitad. Los conectores mas utilizados son los DB9.

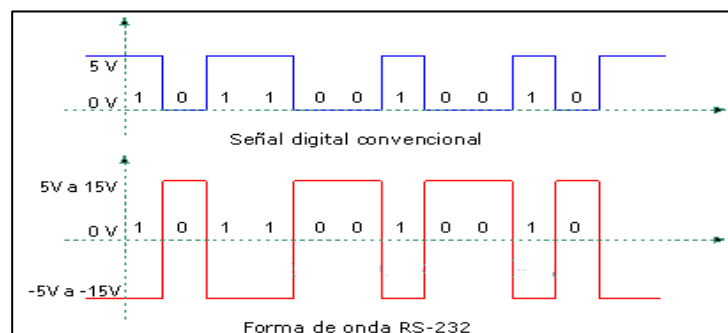
#	Pin	E/S	Función	Conector DB 9
1			Tierra de Chasis	
2	RXD	E	Recibir Datos	
3	TXD	S	Transmitir Datos	
4	DTR	S	Terminal de Datos Listo	
5	SG		Tierra de señal	
6	DSR	E	Equipo de Datos Listo	
7	RTS	S	Solicitud de Envío	
8	CTS	E	Libre para Envío	
9	RI	S	Timbre Telefónico	

Cap.2: Fig. 37 Pines más importantes del conector DB9.

FUENTE: (48).

2.7.1.2 NIVELES LÓGICOS RS-232

En las comunicaciones seriales RS-232 los valores para representar los niveles lógicos (unos y ceros) son muy diferentes de los que se usa en la tecnología TTL, es decir aquí no existe 5voltios para uno y 0voltios para cero **(49)**.

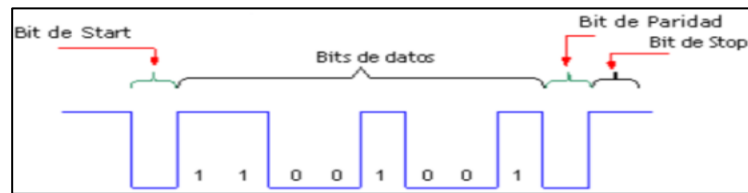


Cap.2: Fig. 38 Niveles de tensión para unos y ceros lógicos

FUENTE: (49)

2.7.1.3 FORMATO DE TRANSFERENCIA DE DATOS

Como en toda comunicación serial, los datos viajan en grupos de bits. En este caso cada grupo o caracter consta de: un bit Start, los bits de Datos (8 por lo general), un bit de Paridad (opcional) y finaliza con uno o dos bits de Stop.



Cap.2: Fig. 39 Formato de un byte de dato en el Estándar RS-232.

FUENTE: (49).

▪ BIT START

Es la transición de 1 a 0 e indica el inicio de una transferencia. En la lógica RS-232 podría significar una transición de -15V a +15V y en la lógica TTL es una transición de 5V a 0V.

▪ BITS DE DATOS

Son los datos que se desean transmitir. Cada dato es de 5, 6, 7 u 8 bits. Casi siempre se ha preferido trabajar con 8 bits (1 byte). El primer bit que se transmite es el menos significativo o LSB (Least Significant Bit).

▪ BIT DE PARIDAD

Este bit es opcional y generalmente se envía después del byte de datos. Sirve para ayudar a detectar posibles errores en las transferencias de datos.

▪ BITS STOP

Los bits de stop son estados de 1 lógico, permite que el receptor sepa el fin de la transmisión de una palabra de datos (un byte).

2.7.1.4 VELOCIDAD DE TRANSMISIÓN

El término velocidad de transmisión se entiende como el número de bits que se transmite por segundo. Debido a que se trata de un tipo de transmisión asíncrona, no existe una señal de reloj que sincronice los bits de datos.

Para que los dispositivos transmisor y receptor se entiendan correctamente, también es necesario que operen a una misma velocidad. Los valores más comunes que fija

el estándar RS-232 son: 1200, 2400, 4800, 9600, 19200, 38400, 57600 y 115200. Es necesario recalcar que las versiones más recientes del estándar RS-232 ponen un límite de 20 kbits, es común emplear los valores altos como 115200 (siempre que sea posible) **(49)**.

2.7.1.5 CONVERSORES USB-RS232

Actualmente existen situaciones donde es necesario convertir o emular un puerto serial RS232 a partir de un puerto USB, la razón de esto es que muchas de las computadoras modernas no incluyen el puerto serial; porque su uso es obsoleto para aplicaciones informáticas. Sin embargo existen muchas aplicaciones en electrónica donde resulta muy conveniente usar el protocolo RS232 por ejemplo, para el intercambio de información entre un dispositivo externo y la computadora.

En el mercado existe una variedad de convertidores de USB a RS232 integrados en un cable o bien como adaptadores usados en proyectos de electrónica.



Cap.2: Fig. 40 Conversores USB-SERIAL.

FUENTE: (50).

Estos adaptadores emular un puerto serial mediante el puerto USB. Poseen un software multiplataforma que una vez instalado crea un puerto serial virtual a través del puerto USB.

2.7.2 PROTOCOLO I2C

La interfaz I2C (Inter- Integrated Circuits) fue inventada por Philips en 1980 para conectar de forma simple componentes electrónicos, implementado en un gran número de circuitos integrados y con licencias otorgadas a más de 50 compañías con un total de 1000 dispositivos compatibles I2C.

Actualmente diseñan dispositivos basados en I2C muchos fabricantes como: Xicor, SGS-Thomson, Siemens, Intel, TI, Maxim, Atmel, Analog Devices, InvenSense, Honeywell, etc.

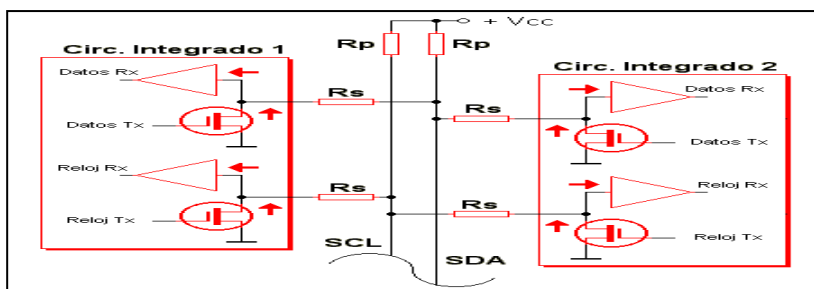
2.7.2.1 CARACTERÍSTICAS

- La comunicación se transmite de forma sincrónica, controlada por una señal de reloj común.
- Está formado por dos hilos o líneas que conectan varios dispositivos mediante un hardware muy simple.
- Posee una comunicación serial, bit a bit. Se transmiten dos señales, una por cada línea y también es necesaria una referencia común de masa.

2.7.2.2 LÍNEAS DE COMUNICACIÓN

SDA (Serial DATA Line) Línea para la transferencia serial de datos.

SCL (Serial CLOCK line) Señal de reloj utilizada para la sincronización de los datos.

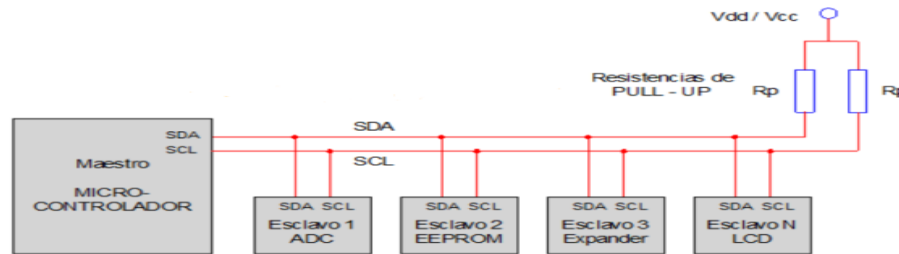


Cap.2: Fig. 41 Conexión I2C.

FUENTE: (41).

Los dispositivos conectados al bus I2C mantienen un protocolo de comunicaciones del tipo maestro/esclavo (master /slave). Las funciones del maestro y del esclavo se diferencian en:

- El circuito maestro inicia y termina la transferencia de información además de, controlar la señal de reloj. Normalmente es un microcontrolador.
- El esclavo es el circuito direccionado por el maestro.



Cap.2: Fig. 42 Topología I2C.

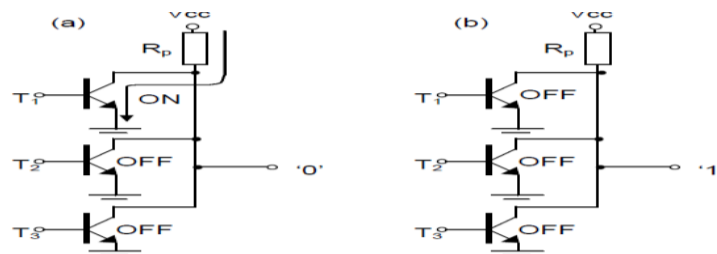
FUENTE: (49).

Las líneas SDA y SCL están conectadas a tensión positiva de alimentación a través de unas resistencias de Pull-Up¹⁴.

La línea SDA es bidireccional, es decir, tanto el maestro como los esclavos actúan como transmisores o receptores de datos, dependiendo de la función del dispositivo. Así por ejemplo, un display es solo un receptor de datos mientras que una memoria recibe y transmite datos.

2.7.2.3 HARDWARE DEL BUS I2C

El hardware del bus I2C se basa en la And cableada. Las etapas de salida de los dispositivos conectados al bus deben ser drenador abierto (CMOS), o colector abierto (TTL), para poder realizar la función And cableada.¹⁵



Cap.2: Fig. 43 AND Cableada (a) una o mas salidas a '0' (b) todas las salidas a '1'.

FUENTE: (51).

¹⁴ Las resistencias pull up son resistores que se conectan entre una señal lógica y el positivo y su función es asegurar que esa señal no quede en un estado flotante. En algunos tipos de dispositivos lógicos, si no se pusieran las resistencias pull up, el estado lógico 1 podría quedar con un valor de tensión intermedio entre cero y uno y confundirse su estado. Son resistores normales, sólo llevan el nombre pull up por la función que cumplen.

¹⁵ La función AND cableada resulta de particular interés cuando se deben combinar muchas entradas, pues se elimina la necesidad de disponer de puertas de muchas entradas. En todos los circuitos de AND cableada se requiere una resistencia externa.

Dependiendo del estado del transistor de salida de cada dispositivo, suele suceder algunos de estos dos casos:

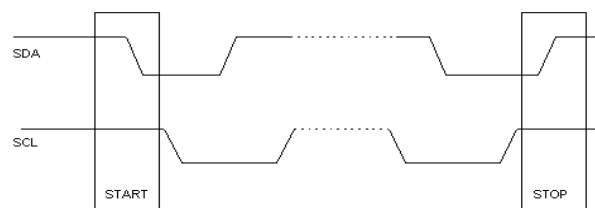
1. Que el transistor esté saturado, con lo cual lleva a nivel bajo ó “0”, a la línea correspondiente, independiente del estado de los otros transistores. Es decir, el bus está ocupado a nivel bajo.
2. Que el transistor esté en corte (estado de alta impedancia) con lo cual el estado de la línea depende de los otros transistores. Es decir, el bus esta libre y si no hay ningún otro transistor saturado, la línea se encuentran en estado alto a través de la resistencia de Pull-Up conectada a la alimentación.

Se transfiere un bit por la línea de datos SDA, y se genera un pulso de reloj por la línea SCL, Los bits de datos transferidos por la línea SDA deben mantenerse estables mientras la línea SCL esté a nivel alto. El estado de la línea SDA solo cambia cuando la línea SCL está a nivel bajo.

Si la línea SDA cambia mientras SCL está a nivel alto, no se interpreta como dato, si no como una condición especial (start o stop), que se explica a continuación.

2.7.2.4 CONDICIONES DE START Y STOP

Para que la transferencia de información pueda ser iniciada el bus no debe estar ocupado; esto quiere decir que los transistores de salida de todos los dispositivos conectados al bus I2C deben estar en alta impedancia. Para indicar que el bus esta libre (no ocupado) las líneas de reloj (SCL) y datos (SDA) deben estar a nivel alto. Una vez que se ha verificado esto, el transmisor procederá a enviar un bit cada pulso de reloj.



Cap.2: Fig. 44 Condición Start y Stop.

FUENTE: (52).

▪ CONDICIÓN DE START

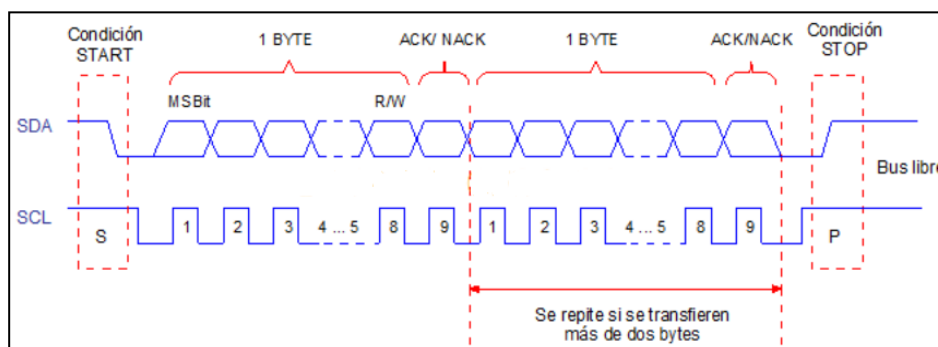
SDA debe estar en flanco de bajada mientras SCL permanece a nivel alto. Esta condición señala el comienzo de la transferencia de datos.

▪ CONDICIÓN DE STOP

SDA en flanco de subida mientras que SCL permanece a nivel alto. Esta es la condición que indica el fin de transferencia.

2.7.2.5 TRANSFERENCIA DE DATOS

Cada dato enviado por la línea SDA debe tener 8 bits (un byte). El número de bytes que es enviado no tiene restricción. El byte de datos se transfiere empezando por el bit 7 que es de mayor peso MSB (Most Significant bit).



Cap.2: Fig. 45 Transferencia de datos sobre el protocolo I2C.

FUENTE: (49).

Una vez que se han transmitido los 8 bits el receptor deberá mandar un bit de asentimiento o reconocimiento (acknowledgement) en el noveno pulso de reloj. El receptor ejecuta este reconocimiento poniendo la señal SDA a un nivel bajo, cada grupo de 8 bits debe ser asentido.

El bit de reconocimiento ACK es obligatorio en la transferencia de datos. El pulso de reloj SCL correspondiente al bit de reconocimiento es generado por el maestro, el transmisor deja libre la línea SDA (la pone en alta impedancia o a nivel alto). El receptor permite que la línea SDA pase a nivel bajo estable durante el período alto del noveno impulso de reloj SCL.

Si el esclavo-receptor que esta direccionado no desea recibir más bytes, el maestro debe detectar la situación y no enviarle más datos. Esto se indica porque el esclavo no genera el bit ACK del último byte quedando la línea SDA a nivel alto, lo cual es detectado por el maestro el cual genera la condición stop y aborta la transferencia de datos.

Si un dispositivo esclavo no recibe o transmite un byte de datos completo en la ejecución de alguna de sus operaciones internas, mantiene la línea SCL a nivel bajo; lo que obliga al maestro a permanecer en un estado de espera, la transferencia de datos se reanudará cuando el dispositivo esclavo se encuentre listo para otro byte de datos y desbloquee la línea de reloj SCL.

2.7.2.6 PASOS PARA ENVIAR O RECIBIR DATOS A TRAVÉS DEL BUS I2C

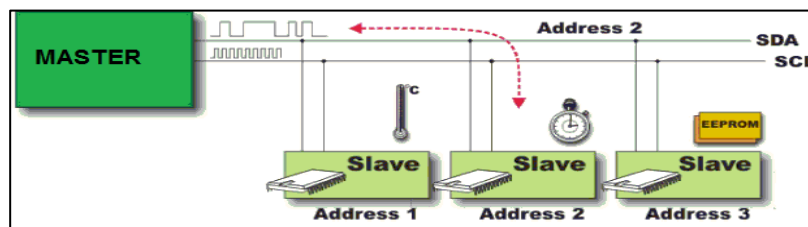
Se deduce que para operar un esclavo sobre el bus I2C son necesarios seis pasos para enviar o recibir información:

1. Un bit de start, que señala el inicio de la transferencia de datos
2. Siete bits de direccionamiento de un esclavo.
3. Un bit de lectura/escritura (R/W) que define si el esclavo es transmisor o receptor.
4. Un bit de reconocimiento ACK (acknowledgement).
5. Un mensaje dividido en bytes (8 bits).
6. Un bit de stop, que indica el fin de la comunicación.

También se debe tomar en cuenta que:

- Los 7 primeros bits del primer byte marcan la dirección del esclavo. El octavo bit (R/W) determina la dirección de los datos:
- Si $R/W=0$, el esclavo es receptor, significa que el maestro escribirá información en el esclavo seleccionado.
- Si $R/W=1$, el esclavo es emisor, significa que el maestro leerá información del esclavo seleccionado.

Cuando un microcontrolador maestro envía una dirección después de la condición de start cada dispositivo comprueba los siete primeros bits de la dirección con la suya propia. El que coincida se considera el dispositivo seleccionado por el maestro, que será un esclavo-receptor o esclavo-emisor dependiendo del bit R/W. Esto quiere decir que cada dispositivo conectado al Bus I2C tiene una única dirección que lo diferencia del resto de circuitos conectados.



Cap.2: Fig. 46 Conexión I2C de varios dispositivos con dirección propia.

FUENTE: (53)

2.7.2.7 TIPOS DE FORMATOS DE TRANSFERENCIA

▪ MAESTRO-EMISOR

Transmite al esclavo-receptor. Si el bit 8 es de escritura (R/W=0).

▪ MAESTRO-RECEPTOR

Lee de un esclavo-emisor inmediatamente después del primer byte. Si el bit 8 es de lectura (R/W=1).

▪ FORMATO COMBINADO

Una transferencia de datos siempre termina con una condición de stop generada por el maestro. Sin embargo, si un maestro todavía desea comunicarse con el bus; genera primero la condición de stop, durante un cambio de dirección dentro de una transferencia, la condición de start y la dirección del esclavo se repiten, pero con el bit R/W invertido. Varias combinaciones de lectura y escritura son posibles dentro de una misma transferencia de datos.

2.7.2.8 APLICACIONES

- Bus de interconexión entre dispositivos en una tarjeta o equipo.
- Sistema de configuración y supervisión en ordenadores servidores.
- Sistemas de gestión de alimentación.
- Conexión en serie de dispositivos externos a un ordenador.
- Tarjetas chip.

2.7.3 PROTOCOLO ZIGBEE

Zigbee es un protocolo de comunicaciones inalámbrico basado en el estándar de comunicaciones para redes inalámbricas IEEE 802.15.4¹⁶. Creado por Zigbee Alliance, una organización con más de 200 grandes empresas (Mitsubishi, Honeywell, Philips, Invensys, entre otras), que trabajaron para crear un sistema estándar de comunicaciones, vía radio y bidireccional, para usarlo dentro de dispositivos de domótica, automatización de edificios (inmótica), control industrial, periféricos de PC, juguetería, sensores médicos. Está específicamente diseñado para remplazar la proliferación de sensores/actuadores individuales.



Cap.2: Fig. 47 Logo ZegBee Alliance.

FUENTE: (54).

2.7.3.1 LA NORMA IEEE 802.15.4

IEEE 802.15.4 es la base sobre la que se define la especificación de ZigBee ya que es un protocolo que define el nivel físico y el control de acceso al medio de las redes LR-WPAN (Low-Rate Wireless Personal Area Network).

¹⁶ IEEE 802.15.4 es un estándar que define el nivel físico y el control de acceso al medio de redes inalámbricas de área personal con tasas bajas de transmisión de datos (low-rate wireless personal area network, LR-WPAN).

ZigBee complementa al protocolo IEEE 802.15.4, su propósito es ofrecer una solución completa construyendo los niveles superiores de la pila de protocolos que IEEE 802.15.4 no cubre.

La particularidad fundamental del protocolo IEEE 802.15.4 entre las WPAN's (Wireless Personal Area Network) es la obtención de costos de fabricación excepcionalmente bajos por medio de la sencillez tecnológica, sin perjuicio de la generalidad o la adaptabilidad, ya que el propósito del protocolo IEEE 802.15.4 es definir los niveles de red básicos para dar servicio a un tipo específico de red WPAN centrada en la habilitación de comunicación entre dispositivos con bajo costo y velocidad.

2.7.3.2 CARACTERÍSTICAS

- ZigBee, es una tecnología inalámbrica con velocidades comprendidas entre 20 kB/s y 250 kB/s y rangos de 10 m a 1200 m. Usa las bandas libres ISM¹⁷ de 2,4 GHz, 868 MHz (Europa) y 915 MHz (EEUU).
- A pesar de coexistir en la misma frecuencia con otro tipo de redes como WiFi o Bluetooth su desempeño no se ve afectado, esto debido a su baja tasa de transmisión y a características propias del estándar IEEE 802.15.4.1
- Capacidad de operar en redes de gran densidad, esta característica ayuda a aumentar la confiabilidad de la comunicación, ya que entre más nodos existan dentro de una red, entonces, mayor número de rutas alternas existirán para garantizar que un paquete llegue a su destino.
- Cada red ZigBee tiene un identificador de red único, lo que permite que coexistan varias redes en un mismo canal de comunicación sin ningún problema. Teóricamente existen hasta 16000 redes diferentes en un mismo canal y cada red esta constituida por hasta 65000 nodos, obviamente estos límites se ven

¹⁷ ISM (Industrial, Scientific and Medical) son bandas reservadas internacionalmente para uso no comercial de radiofrecuencia electromagnética en áreas industrial, científica y médica.

truncados por algunas restricciones físicas (memoria disponible, ancho de banda, etc.).

- Es un protocolo de comunicación multi-salto, es decir, establece comunicación entre dos nodos aún cuando estos se encuentren fuera del rango de transmisión, siempre y cuando existan otros nodos intermedios que los interconecten, de esta manera, se incrementa el área de cobertura de la red.
- Su topología de malla (MESH) permite a la red auto-recuperarse de problemas en la comunicación aumentando su confiabilidad.

2.7.3.3 ARQUITECTURA

2.7.3.3.1 MODELO DE REFERENCIA ISO/OSI

El estándar IEEE 802 define únicamente dos capas fundamentales: el Sistema Abierto de Interconexiones OSI y la Organización Internacional de Normalización ISO, las cuales son:

- Capa física PHY (Physical Layer).
- Capa de Enlace de Datos o MAC

Las otras capas no se especifican en el estándar y son normalmente descritas por el consorcio industrial formado por compañías interesadas en la fabricación y uso del estándar en particular.

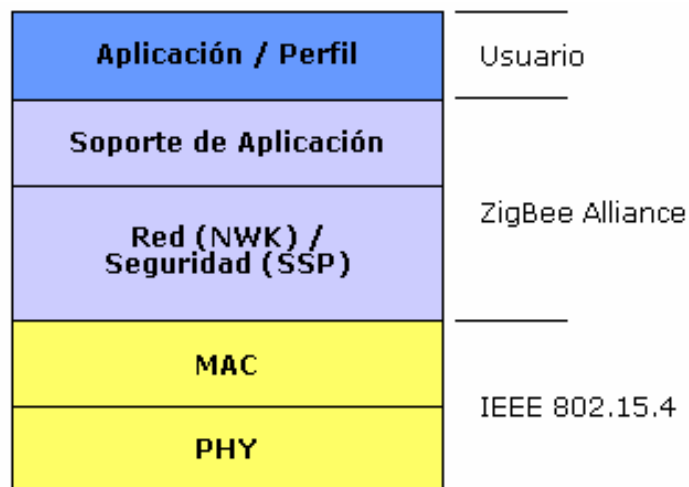
Para IEEE 802.15.4, Alliance ZigBee dirige el desarrollo de las capas superiores, por medio de la definición del perfil de aplicación. Estos perfiles hacen uso de un modelo de referencia simplificado de cinco capas de la ISO/OSI.

MODELO OSI-ISO	MODELO OSI-ISO SIMPLIFICADO	MODELO IEEE 802
7. Aplicación	Aplicación	Capas superiores
6. Presentación	Perfil de aplicación	
5. Sesión		
4. Transporte		
3. Red	Red	Control de enlace lógico (MAC)
2. Enlace de datos	Enlace de datos	Control de acceso al medio (MAC)
1. Física	Física	Física (PHY)

Cap.2: Tabla. 5 Modelo ISO/OSI y Modelo estándar IEEE 802.

FUENTE: (55)

Para finalmente obtener esta estructura donde se ven las capas que maneja el estándar 802.15.4 y las capas superiores que se le atribuyen a la Zigbee Alliance. En Fig.48 se muestran las diferentes capas que conforman la pila de protocolos para Zigbee.



Cap.2: Fig. 48 Arquitectura ZigBee.

FUENTE: (56).

2.7.3.3.2 CAPAS ARQUITECTURA ZIGBEE

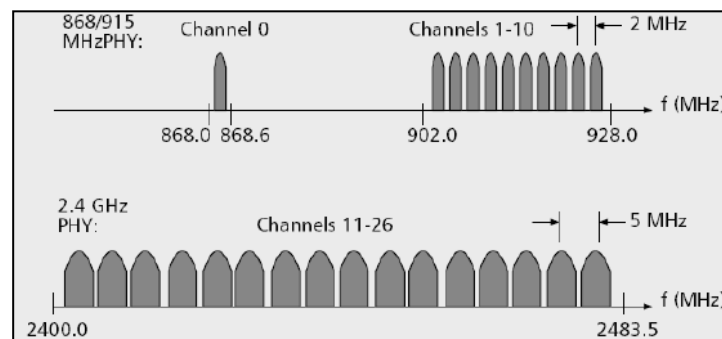
A) CAPA FÍSICA

La capa física (PHY) provee la interfaz con el medio físico donde ocurre la comunicación. Esta capa es el primer componente del modelo ISO/OSI y se encarga de:

- Control (activación, desactivación) de transmisor-receptor y actuadores.
- Detección de energía (dependiendo del transmisor-receptor).
- Calidad del enlace.
- Asignación de canales.
- Selección de canales.
- Medición de variables.
- Transmisión y recepción de los paquetes de mensajes a través del medio.

CANALES DE OPERACIÓN.

Se especifica un total de 27 canales por medio de las tres bandas de frecuencia. Los canales se enumeran desde 0 a 26.



Cap.2: Fig. 49 Estructura de canales de operación Zigbee.

FUENTE: (56).

B) CAPA DE ENLACE DE DATOS MAC (MEDIUM ACCESS CONTROL)

El estándar IEEE 802.15.4 usa el algoritmo de Acceso múltiple con un mecanismo que evita las colisiones de datos, el cual chequea la disponibilidad del canal antes de

transmitir y así evitar colisiones con otros transmisores. También proporciona la ayuda para desarrollar tres tipos de topologías inalámbricas las cuales son: Topología en Estrella, Topología en Árbol, Topología en Malla.

La subcapa MAC se encarga de las siguientes funciones:

- Generación de tramas de acuse de recibo (acknowledgment frames),
- Asociación / disociación
- Control de seguridad.

C) CAPA DE RED

La capa de red (NWK), permite el correcto uso del subnivel MAC y ofrecer una interfaz adecuada para su uso por parte de la capa de aplicación. En esta capa se brindan los métodos necesarios para: iniciar la red, unirse a la red, enrutar paquetes dirigidos a otros nodos en la red, proporcionar los medios para garantizar la entrega del paquete al destinatario final, filtrar paquetes recibidos, cifrarlos y autentificarlos. Cuando esta capa se encuentra cumpliendo la función de unir o separar dispositivos a través del controlador de red, ejecuta la implementación de seguridad, y encamina tramas a sus respectivos destinos; además es responsable de crear una nueva red y asignar direcciones a los dispositivos de la misma.

D) CAPA DE SOPORTE DE LA APLICACIÓN

Esta capa es la responsable de mantener el rol con el que el nodo: filtra paquetes a nivel de aplicación, mantiene la relación de grupos y dispositivos con los que la aplicación interactúa y simplifica el envío de datos a los diferentes nodos de la red. La capa de Red y de soporte a la aplicación es definida por la ZigBee Alliance.

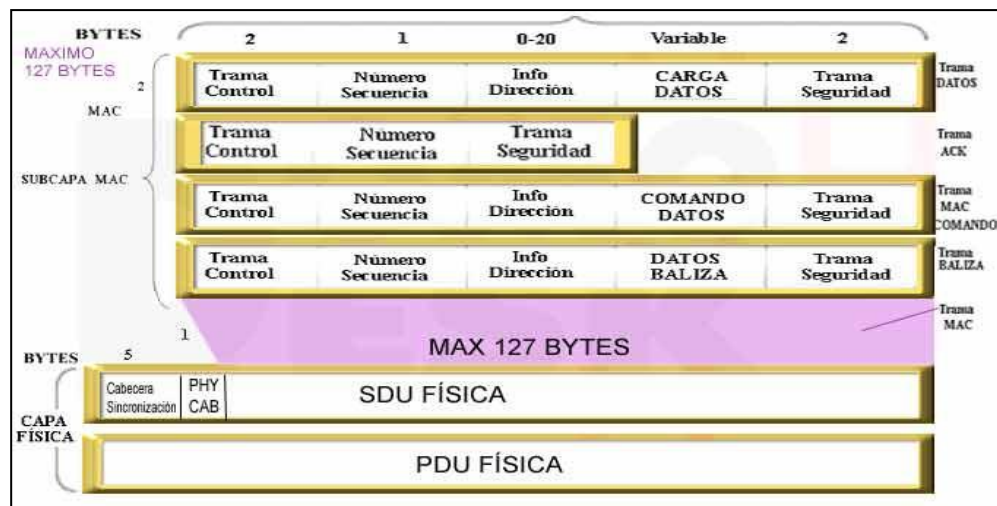
E) CAPA DE APLICACIÓN.

En el nivel conceptual más alto se encuentra la capa de aplicación que no es otra cosa que la aplicación misma y de la que se encargan los fabricantes. Es en esta capa donde se encuentran los ZDO (ZigBee Device Objects) que se encargan de definir el papel del dispositivo en la red, si el actuará como coordinador, ruteador o

dispositivo final; la subcapa APS y los objetos de aplicación definidos por cada uno de los fabricantes.

2.7.3.4 EMPAQUETAMIENTO Y DIRECCIONAMIENTO

En ZigBee, el empaquetamiento se efectúa en cuatro tipos diferentes de paquetes, los cuales son: datos, ACK, MAC y baliza.



Cap.2: Fig. 50 Campos de los cuatro tipos de paquetes básicos de ZigBee.

FUENTE: (56).

El paquete de datos tiene una carga de hasta 104 bytes. La trama está numerada para asegurar que todos los paquetes llegaron a su destino. Un campo asegura que el paquete se ha recibido sin errores. Esta estructura aumenta la fiabilidad en condiciones complicadas de transmisión **(57)**.

La estructura de los paquetes ACK, llamada también paquete de reconocimiento, es donde se ejecuta una realimentación desde el receptor al emisor, de esta manera se confirma que el paquete se ha recibido sin errores. Se incluye un tiempo de silencio entre tramas, para enviar un pequeño paquete después de la transmisión de cada paquete **(57)**.

El paquete MAC, se utiliza para el control remoto y la configuración de dispositivos/nodos. Una red centralizada utiliza este tipo de paquetes para configurar la red a distancia **(57)**.

El paquete baliza se encarga de “despertar” los dispositivos que “escuchan” y luego vuelven a “dormirse” si no reciben nada más. Estos paquetes son importantes para mantener todos los dispositivos y los nodos sincronizados, sin tener que gastar una gran cantidad de batería estando todo el tiempo encendidos **(56)**.

2.7.3.5 DISPOSITIVOS QUE CONSTITUYEN UNA RED ZIGBEE

En una red ZigBee se encuentran y detectan tipos de dispositivos diferentes: según el papel y según su funcionalidad que desarrollen en la red.

A) SEGÚN EL PAPEL QUE DESARROLLEN EN LA RED

▪ COORDINADOR ZIGBEE (ZIGBEE COORDINATOR, ZC)

Consiste en el dispositivo más completo de los tres, puesto que sus funciones son las de controlar y coordinar la red y los caminos que deben seguir los dispositivos para conectarse entre ellos. Se encuentra obligatoriamente un ZC en cada red ZigBee.

▪ ROUTER ZIGBEE (ZIGBEE ROUTER, ZR)

Su función es la de interconectar los dispositivos separados en la topología de la red, además de ofrecer un nivel de aplicación para la ejecución de código de usuario.

▪ DISPOSITIVO FINAL (ZIGBEE END DEVICE, ZED)

En este dispositivo están representadas las principales características de ZigBee, como son el bajo consumo y el bajo costo. Los ZED poseen la funcionalidad necesaria para comunicarse con su nodo padre, que suelen ser el Router ZigBee o el Coordinador ZigBee, pero no le es permitido transmitir información destinada a otros dispositivos. Es por ello, que este tipo de dispositivo acostumbra estar

“dormido”¹⁸ la mayor parte del tiempo aumentando así la vida media de sus baterías. Un ZED tiene requerimientos mínimos de memoria y es por ello significativamente más barato.

B) SEGÚN LA FUNCIÓN QUE DESARROLLEN EN LA RED

▪DISPOSITIVO DE FUNCIONALIDAD COMPLETA (FFD, FULL-FUNCTION DEVICE)

Conocido también como nodo activo. Gracias a la memoria adicional y a la capacidad de computar funcionan como: Coordinador o Router ZigBee de una red de área personal (PAN) o como un nodo normal. Implementa un modelo general de comunicación que le permite establecer un intercambio con cualquier otro dispositivo pudiendo encaminar mensajes, en cuyo caso se le denomina coordinador (coordinador de la PAN si es el responsable de toda la red y no sólo de su entorno).

▪DISPOSITIVO DE FUNCIONALIDAD REDUCIDA (RFD, REDUCED-FUNCTION DEVICE)

Conocido también como nodo pasivo. Posee una capacidad y funcionalidad limitada para garantizar un bajo costo y una gran simplicidad, por ello sólo se comunican con FFD's y nunca pueden ser coordinadores. Básicamente constituyen los sensores de la red **(55)**.

2.7.3.6 MODOS DE FUNCIONAMIENTO DE ZIGBEE

El funcionamiento de ZigBee debe cumplir la premisa del bajo consumo de sus nodos. Para ello un nodo ZigBee, tanto activo como pasivo, reduce su consumo gracias a que permanece “dormido” la mayor parte del tiempo, incluso muchos días seguidos. Cuando se requiere su uso, el nodo ZigBee es capaz de despertar en un

¹⁸ Cuando se dice que el nodo permanece “dormido” se refiere a que está a la espera de ser activado por parte del Router o del Coordinador ZigBee.

tiempo ínfimo, para volverse a “dormir” cuando deje de ser requerido. El tiempo que tarda un nodo cualquiera en despertarse es de aproximadamente 15ms.

2.7.3.7 ESTRATEGIAS DE CONEXIÓN EN UNA RED ZIGBEE

En las redes ZigBee, se usan dos modos de funcionamiento diferentes: con balizas o sin balizas.

▪ CON BALIZAS

Es un mecanismo de control del consumo de potencia en la red. Permite a todos los dispositivos saber cuándo transmitir. Las balizas que dan nombre a este tipo de entorno, se usan para poder sincronizar todos los dispositivos que conforman la red, identificando la red domótica, y describiendo la estructura de la “supertrama”¹⁹. Los intervalos de las balizas son asignados por el coordinador de red y varían desde los 15ms hasta los 4 minutos.

Este modo es más recomendable cuando el coordinador de red trabaja con una batería. Los dispositivos que conforman la red, escuchan a dicho coordinador durante el “balizamiento” (envío de mensajes a todos los dispositivos -broadcast-, entre 0,015 y 252 segundos). Un dispositivo que quiera intervenir, lo primero que tendrá que hacer es registrarse para el coordinador, y es entonces cuando mira si hay mensajes para él. En el caso de que no haya mensajes, este dispositivo vuelve a “dormir”, y se despierta de acuerdo a un horario que ha establecido previamente el coordinador. En cuanto el coordinador termina el “balizamiento”, vuelve a “dormirse”.

▪ SIN BALIZAS

Se usa el acceso múltiple al sistema Zigbee en una red punto a punto cercano. Donde, cada dispositivo es autónomo, pudiendo iniciar una conversación, en la cual

¹⁹ Una supertrama está formada por dieciséis slots de igual capacidad, que pueden dividirse en una parte activa y otra pasiva, en la que el coordinador puede ahorrar energía ya que no tendrá que realizar labores de control. Están comprendidas entre dos balizas y proveen sincronización e información de configuración a otros dispositivos.

los otros pueden interferir. A veces, ocurre que el dispositivo destino no oye la petición, o el canal está ocupado.

Este método se usa típicamente en los sistemas de seguridad, en los cuales sus dispositivos (sensores, detectores de movimiento o de rotura de cristales), duermen prácticamente todo el tiempo (el 99,999%). Para que se les tenga en cuenta, estos elementos se “despiertan” de forma regular para anunciar que siguen en la red. Cuando se produce un evento (en el sistema será cuando se detecta algo), el sensor “despierta” instantáneamente y transmite la alarma correspondiente. Es en ese momento cuando el coordinador de red, recibe el mensaje enviado por el sensor, y activa la alarma. En este caso, el coordinador de red se alimenta de la red principal durante todo el tiempo **(58)**.

2.7.3.8 ACCESO AL MEDIO

El medio físico es un recurso al que se accede utilizando CSMA/CA. Las redes que no utilizan las balizas hacen uso de una variación del mismo basada en la escucha del medio, temporizada por un algoritmo de backoff²⁰, salvo en el caso de las confirmaciones (ACK, Acknowledgement).

Estos mensajes de confirmación suelen ser opcionales en algunos casos. La recepción de una confirmación certifica el éxito en el envío. En cualquier caso, si un dispositivo es incapaz de procesar una trama en un momento dado, no se confirma su recepción. Se hacen reintentos basados en timeout²¹ un cierto número de veces, tras lo cual se decide: si seguir intentándolo o considerarlo como un error de transmisión.

²⁰ El algoritmo de Backoff da un número aleatorio y entero de ranuras temporales (slot time) y su función es la de reducir la probabilidad de colisión que es máxima cuando varias estaciones están esperando a que el medio quede libre para transmitir. Cuando el medio se detecta como ocupado antes de la transmisión, después de cada retransmisión y después de cada transmisión correcta.

²¹ Timeout define, en segundos, el tiempo que el servidor esperará por recibir y transmitir durante la comunicación. Timeout está configurado por defecto a 300 segundos, lo cual es apropiado para la mayoría de las situaciones.

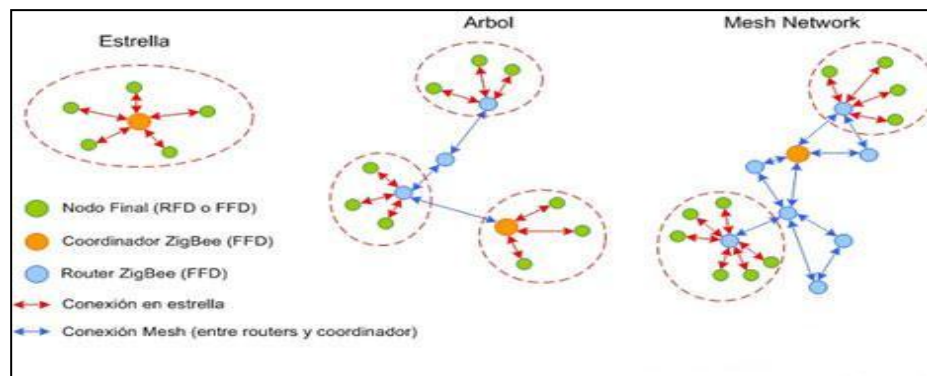
2.7.3.9 ARQUITECTURA DE SEGURIDAD

ZigBee utiliza claves de 128 bits en sus mecanismos de seguridad. Una clave se asocia a una red (utilizable por los niveles de ZigBee y el subnivel MAC) o a un enlace (en tal caso, adquirida por preinstalación, acuerdo o transporte). Las claves de enlace se establecen en base a una clave maestra que controla la correspondencia entre claves de enlace. Como mínimo la clave maestra inicial debe obtenerse por medios seguros (transporte o preinstalación), ya que la seguridad de toda la red depende de ella.

2.7.3.10 TOPOLOGÍA DE REDES DE ZIGBEE

Un aspecto a tener muy en cuenta son los tipos de topologías de red que permite el estándar que soporta ZigBee. Su nivel de red permite tres topologías distintas:

- Topología en estrella.
- Topología en árbol (CLUSTER TREE).
- Topología en malla.



Cap.2: Fig. 51 Topologías Estrella, Árbol y Malla de una red Zigbee.

FUENTE: (59).

2.7.3.11 COMPARATIVA DE TECNOLOGÍAS WIRELESS

Hasta la actualidad, se han creado nuevas redes Wireless, como Wi-Fi, Bluetooth, y otras venideras WiMAX, USB inalámbrico, etc. En la siguiente tabla se efectúa una comparativa de las tres tecnologías más conocidas y en proceso de expansión. Las

cámaras Wireless, destacadas por el control remoto, son un ejemplo de cómo se logra aplicar estas tecnologías para la domótica y el control de áreas.

Estándar	Ancho de Banda	Consumo de Potencia	Ventajas	Aplicaciones
Wi-Fi	Hasta 54Mbps	400mA transmitiendo, 20ma en reposo	Gran ancho de banda	Navegar por Internet, redes de ordenadores, transferencia de ficheros
Bluetooth	1Mbps	40mA transmitiendo, 0.2mA en reposo	Interoperatividad, sustituto del cable	Wireless USB, móviles, informática casera
ZigBee	250Kbps	30mA transmitiendo, 3mA en reposo	Batería de larga duración, bajo costo	Control remoto, productos dependientes de la batería, sensores , juguetería

Cap.2: Tabla. 6 Comparativa de tecnologías Wireless.

FUENTE: (56).

2.7.3.12 VENTAJAS DE LAS REDES ZIGBEE

- Ideal para conexiones punto-punto y punto-multipunto.
- Diseñado para el direccionamiento de información y el refrescamiento de la red.
- Opera en la banda libre de ISM 2.4 Ghz para conexiones inalámbricas.
- Óptimo para redes de baja tasa de transferencia de datos.
- Alojamiento de 16 bits a 64 bits de dirección extendida.
- Reduce tiempos de espera en el envío y recepción de paquetes.
- Bajo consumo de energía.
- Soporte de múltiples topologías de red: estática, dinámica, estrella y malla.
- Una red ZigBee consta de más de 65000 nodos distribuidos en subredes de 255 nodos.
- Son más baratos y de construcción más sencilla.

2.7.3.13 DESVENTAJAS DE LAS REDES ZIGBEE

- La tasa de transferencia es muy baja.
- Solo manipula textos pequeños comparados con otros estándares inalámbricos.
- ZigBee trabaja de manera que no es compatible con Bluetooth en todos sus aspectos porque no llegan a tener las mismas tasas de transferencia, ni la misma capacidad de soporte para nodos.
- Tiene menor cobertura porque pertenece a redes inalámbricas de tipo WPAN.

2.7.3.14 APLICACIONES

ZigBee se aplica al mercado donde no se requiere que las tasas de transmisión sean altas, las cuales comprenden una amplia variedad de aplicaciones, las mismas están determinadas por unas 300 compañías que conforman la alianza ZigBee; un gran número de ellas se encuentra desarrollando productos que van desde electrodomésticos hasta teléfonos celulares. Al usar esta tecnología no se tiene problemas en la instalación del cableado debido a que es una tecnología inalámbrica. Otra de las aplicaciones que ha tomado fuerza, son los sistemas de medición avanzada, medidores de agua, luz y gas que forman parte de una red, con otros dispositivos como displays, ubicados dentro de las casas; que monitorean el consumo de energía. También interactúan con electrodomésticos o cualquier otro sistema eléctrico como bombas de agua o calefacción, con la finalidad de aprovechar mejor la energía.



Cap.2: Fig. 52 Aplicaciones Zigbee.

FUENTE: (56)

2.8 ARDUINO

Arduino es una plataforma Open Source Hardware, basada en una placa con un microcontrolador y un entorno de desarrollo, para la elaboración de proyectos de electrónica. El proyecto inició en Italia (Ivrea) en el año 2005 por Massimo Banzi y David Cuartielles, cuyo destino es la construcción de proyectos de interacción. Incluye un entorno de desarrollo que implementa el Lenguaje Processing²² basado en Wiring.²³

Arduino siente el entorno mediante la recepción de entradas desde una variedad de sensores y afecta a su alrededor mediante el control de luces, motores y otros artefactos

Las placas se ensamblan a mano o se adquieren pre ensambladas, el software se descarga gratuitamente. Los diseños de referencia del hardware (archivos CAD) están disponibles bajo licencia Open-Source, de descarga libre.

2.8.1 CARACTERÍSTICAS

▪ BAJO COSTO

Relativamente baratas, un promedio de \$35 el módulo ensamblado. Teniendo la posibilidad de obtener la versión para ensamblar a mano.

▪ MULTIPLATAFORMA

El software de Arduino se ejecuta en sistemas operativos Windows, Macintosh OSX y GNU/Linux. Considerando que la mayoría de los microcontroladores están limitados a Windows.

²² Processing Es un lenguaje de programación de código abierto, para crear imágenes, animaciones e interacciones. Inicialmente desarrollado para servir como un software como block de dibujo y para enseñar los fundamentos de la programación de computadora en un contexto visual.

²³ Wiring es un entorno de programación de código abierto y una tarjeta electrónica diseñada para llevar el arte a la electrónica, usando un medio tangible, para la enseñanza y el aprendizaje de la programación y creación de prototipos con electrónica. Se ilustra el concepto de la programación con la electrónica a un nivel físico de control de hardware que son necesarios para explorar el diseño de interacción física y los aspectos a través de medios tangibles.

▪ ENTORNO DE PROGRAMACIÓN SIMPLE Y CLARO

Fácil de usar para principiantes, y suficientemente flexible para que usuarios avanzados puedan aprovecharlo al máximo. Basado en entorno de programación Processing.

▪ SOFTWARE DE CÓDIGO ABIERTO

Arduino es una herramienta de código abierto, disponible para extensión por programadores experimentados. El lenguaje de programación puede ser expandido mediante librerías C++, también se puede pasar desde Arduino a la programación AVR GCC y añadir directamente este código a programas Arduino.

▪ HARDWARE DE RECURSO ABIERTO

Está basado en microcontroladores ATMEGA328P de Atmel. Los planos para los módulos están publicados bajo licencia Creative Commons²⁴, permitiendo desarrollar una versión propia del módulo. A necesidad del usuario.

2.8.2 HARDWARE ARDUINO

Existen múltiples versiones de la placa Arduino actualmente en su mayoría basadas en los ATMEGA 328.

▪ DIECIMILA

Esta placa se conecta al computador con un cable estándar USB y contiene todo lo que se necesita para programar y usar esta placa. Es ampliada con variedad de dispositivos: placas hijas con características específicas.

²⁴ Las licencias Creative Commons o CC están inspiradas en la licencia GPL (General Public License) de la Free Software Foundation, compartiendo buena parte de su filosofía. La idea principal detrás de ellas es posibilitar un modelo legal ayudado por herramientas informáticas, para así facilitar la distribución y el uso de contenidos.

▪ NANO

Una placa compacta diseñada para uso como tabla de pruebas, Nano se conecta al computador usando un cable USB Mini-B.

▪ BLUETOOTH

El Arduino BT contiene un módulo bluetooth que permite comunicación y programación sin cables. Es compatible con los dispositivos Arduino.

▪ LYLTY PAD

Diseñado para aplicaciones como prendas de vestir, usa complementos similares como fuente de alimentación, sensores, actuadores unidos por hilo conductor. La placa esta basada en el Atmega168 o el ATmega328.

▪ MINI

Esta es la placa más pequeña de Arduino. Trabaja bien en tabla de pruebas o en aplicaciones en la que prima el espacio. Se conecta al computador usando el cable Mini USB. Es muy frágil para el propósito de este proyecto.

▪ SERIAL

Es una placa básica que usa RS232 como una interfaz con el ordenador para programación y comunicación. Fácil de usar en ejercicios de aprendizaje.

▪ SERIAL SINGLE SIDED

Esta placa esta diseñada para ser grabada y ensamblada a mano. Es ligeramente más grande que la Diecimila, pero compatible con los Shields.²⁵

²⁵ Los Shields son placas que se colocan sobre la placa Arduino y amplían una nueva función para que sea controlada desde Arduino, para controlar diferentes aparatos, adquirir datos, etc. Entre los shields más destacados están: xBee Shield, Motor Shield, GPS Shield, WiFly Shield, Ethernet Shield.

2.8.3 SOFTWARE ARDUINO

El software de Arduino consiste en un entorno de desarrollo (IDE) y un núcleo de bibliotecas. El IDE está escrito en Java y las bibliotecas centrales están escritos en C y C ++.

Arduino recibe datos del exterior a través de sensores y luego ejecuta una acción con luces, motores, etc. Esto se logra usando el lenguaje de programación propio de Arduino (Arduino Programming Language).

AVR Libc

Los programas compilados con Arduino se enlazan con AVR Libc, de tal manera que tienen acceso a algunas de sus funciones. El AVR Libc es un proyecto de software libre cuyo objetivo es proporcionar una biblioteca en C de alta calidad, para utilizarse con el compilador AVR GCC sobre microcontroladores Atmel AVR.²⁶

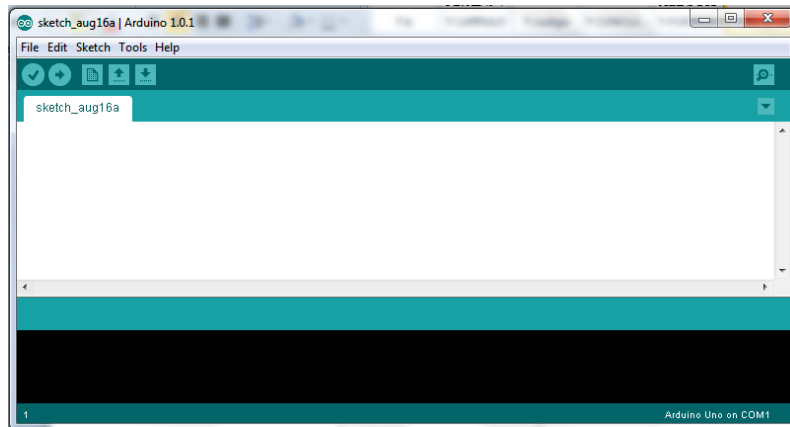
AVR Libc se compone de 3 partes:

- avr-binutils
- avr-gcc
- avr-libc

2.8.3.1 ENTORNO ARDUINO

La interfaz de Arduino es sencilla y su uso es amigable para el programador permitiéndole fácilmente familiarizarse con ella. Cuenta con opciones de menú: reproducir, parar, crear nuevo, abrir, guardar y exportar.

²⁶ AVR Libc Home Page, 05-01-2012 <http://www.nongnu.org/avr-libc/>.



Cap.2: Fig. 53 Entorno de programación Arduino.

FUENTE: (60).

▪ SKETCH

Verify/Compile. Compila y verifica errores de sintaxis.

Import Library. Añade las cabeceras necesarias para la elaboración de un programa específico.

Show Sketch Folder. Abre las carpetas en donde residen las librerías de Arduino, siendo estas creadas por el usuario o propias de Arduino.

Add File. Añade otro archivo.

▪ TOOLS

Auto Format. Formatea el código apropiadamente. Ejecuta una sangría en el texto del código de manera que las llaves de apertura y cierre estén alineadas y que las órdenes o instrucciones estén dentro de estas llaves y con más sangría.

Copy for Discourse. Copia el código de la rutina al portapapeles para subirlo a un foro.

Board. Selecciona la placa Arduino que se usará para controlar y compilar la rutina.

Serial Port. Permite elegir los puertos reales o virtuales de la computadora.

Burn Bootloader. Los elementos que contiene este menú permiten grabar un bootloader (gestor de arranque) en la placa Arduino. Para grabar un bootloader con el AVR ISP se selecciona el elemento que corresponde al puerto serial.

2.8.3.2 BOOTLOADER

Bootloader es un gestor de arranque que permite inicializar todos los recursos de un sistema cuando es energizado. Es el primer programa que se ejecuta para dar paso al programa principal (registros, sentencias de programa, programas de inicio).

En la placa Arduino el microcontrolador Atmega 328P contiene precargado el bootloader, el cual carga los bits y registros necesarios para que se pueda ejecutar el programa que quedó en el microcontrolador cargado la última vez.

A través del uso del bootloader y por medio de un cable USB se programa el Atmega 328P que es parte de Arduino Nano V3, sin necesidad de usar un programador externo.

2.8.3.3 ESTRUCTURA DE UN PROGRAMA EN ARDUINO


La estructura básica para la programación mediante Arduino está compuesta de dos partes y estas encierran bloques que contienen declaraciones e instrucciones.

setup()

Se constituye como la primera función a ejecutar en el programa. Es usada para llamar a funciones de configuración ya que se ejecuta una sola vez, por ejemplo, asignar la funcionalidad de un pin a través de la función `pinMode` o también inicializar la comunicación serial por medio de `Serial.begin`.

loop()

Esta función se ejecuta indefinidamente luego de `setup()`, lee las entradas y activa las salidas. Efectúa la mayor parte del trabajo, siendo el núcleo de los programas Arduino **(61)**.

The image shows a screenshot of the Arduino IDE interface. The window title is "parpadeoLed | Arduino 1.0". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". The main text area contains the following code:

```
#define PIN 10

void setup(){
  Serial.begin(38400); //inicializa la comunicacion serial a 38400 baudios
  pinMode(PIN,OUTPUT); // definimos al pin 10 como salida
}

void loop(){
  digitalWrite(PIN,HIGH); //escribe un 1 logico en el pin 10
  delay(1000); //espera un segundo
  digitalWrite(PIN,LOW); //escribe un 0 logico en el pin 10
  delay(1000); //espera un segundo
}
```

Below the code area, there is a status bar that says "Done Saving". At the bottom of the window, it displays "Binary sketch size: 2378 bytes (of a 30720 byte maximum)" and "13" on the left, and "Arduino Nano w/ATmega328 on COM6" on the right.

Cap.2: Fig. 54 Programa parpadeo de un led.

FUENTE: AUTOR.

En Fig.54, se presenta un sketch el cual hace parpadear un led que esta conectado en el pin 10 del módulo Arduino cada segundo.

CAPÍTULO 3

3. ENTORNO DE DESARROLLO Y MATERIALES

La elección del hardware, software y materiales necesarios para el desarrollo del proyecto fue crucial, de tal manera que se realizó una investigación minuciosa para elegir los dispositivos y el entorno de desarrollo donde estos trabajarían.

Luego de analizar las características y utilidades de los diferentes elementos electrónicos existentes en el mercado, se eligió los que por sus bondades permiten utilizar herramientas Open Source Software y Open Source Hardware. Estas herramientas trabajan bajo la plataforma Arduino, la cual se expuso en el capítulo dos.

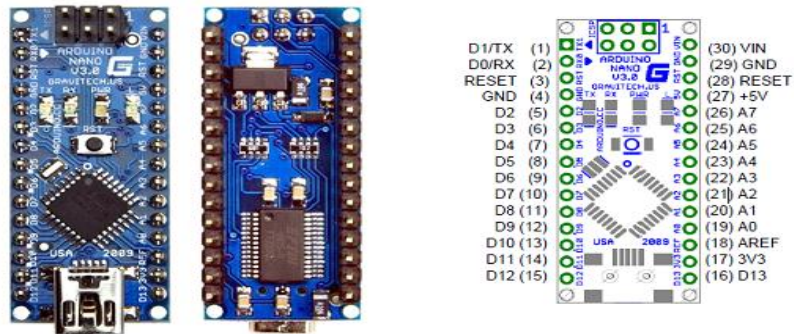
Los materiales usados para la elaboración del “Guante Electrónico de Datos”, reúnen las particularidades necesarias, en lo que se refiere a la comodidad y flexibilidad, sin descuidar el aporte técnico, que en este caso es la parte electrónica de cada elemento.

En este capítulo se detalla el hardware, software y los materiales usados en la construcción del “Guante Electrónico de Datos”.

3.1 HARDWARE

3.1.1 ARDUINO NANO USB MICROCONTROLLER V3.

El Arduino Nano USB es una pequeña y completa placa basada en el microcontrolador ATmega328P.



Cap.3: Fig. 1 ARDUINO NANO USB MICROCONTROLLER V3.

FUENTE: (62).

Pin No.	Nombre	Tipo	Descripción
1-2, 5-16	D0-D13	I/O	Salida y entrada digital puerto 0 - 13
3, 28	RESET	Input	Reset (activado en bajo)
4, 29	GND	PWR (led indicador)	Sumistro de tierra
17	3V3	Output	+3.3V salida (para FTDI)
18	AREF	Input	Referencia ADC
19-26	A0 – A7	Input	Entrada analógica
27	+5v	Output o Input	+5V de salida del regulador interno +5V de entrada desde una alimentación externa
30	VIN	PWR (led indicador)	Sumistro de voltaje

Cap.3: Tabla. 1 Designación de pines Arduino Nano V3.

FUENTE: (62).

3.1.1.1 DESCRIPCIÓN

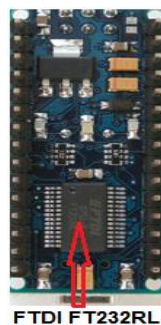
- Microcontrolador Atmel ATmega328P.
- Tensión de Operación (nivel lógico) 5 V.
- Tensión de Entrada (recomendado) 7-12 V.
- Tensión de Entrada (límites) 6-20 V
- Pines E/S Digitales 14 (de los cuales 6 proveen de salida PWM)
- Corriente máxima por cada PIN de E/S 40 mA.
- Entradas Analógicas: 8.

- Frecuencia de reloj: 16 MHz.
- Memoria: 32 KB de los cuales 2KB son usados por el bootloader.
- SRAM: 2 KB / EEPROM: 1 KB.
- Conversor FTDI FT232RL.

3.1.1.2 FTDI

FTDI (Future Technology Devices International) fabricante de chips conversores USB-serial. Permite que un puerto serial maneje dispositivos con interfaz USB trabajando como esclavo (familia FT232, FT245, FT2232) o también como maestro (familia VINCULUM). FTDI proporciona conectividad en diferentes formatos: SMD, DIP y cable, además permite que cada cliente pueda seleccionar el producto al que mejor se adapte.

Los drivers para el chip FTDI incluidos en el software Arduino proporcionan al software del computador un puerto de comunicación virtual. El arduino Nano V3 posee un conversor FTDI FT232RL.



Cap.3: Fig. 2 FTDI ubicado en el lado inferior del Arduino Nano V3.

FUENTE: (50).

3.1.1.3 ENTRADA Y SALIDA

Cada uno de los 14 pines digitales del Arduino Nano se usan como entrada o salida, usando las funciones `pinMode()`, `digitalWrite()`, y `digitalRead()`; operan a 5 voltios. Cada pin provee o recibe un máximo de 40mA y poseen una resistencia de pull-up (desconectada por defecto) de 20 a 50 kOhmios.

Además algunos pines poseen funciones especializadas:

▪ **SERIAL**

D0 (RX) y D1 (TX). (RX) usado para recibir y (TX) usado para transmitir datos TTL vía serial. Estos pines están conectados a los pines correspondientes del chip USB-a-TTL de FTDI.

▪ **INTERRUPCIONES EXTERNAS**

Pines D2 y D3. Estos pines son configurados para activar una interrupción por paso a nivel bajo, por flanco de bajada o flanco de subida, o por un cambio de valor.

▪ **PWM**

Pines D3, D5, D6, D9, D10, y D11. Proveen de una salida PWM de 8-bits cuando se usa la función `analogWrite()`.

▪ **SPI**

Pines D10 (SS), D11 (MOSI), D12 (MISO), D13 (SCK). Estos pines soportan la comunicación SPI, la cual, a pesar de poseer el hardware, no está actualmente soportada en el lenguaje Arduino.

▪ **LED**

Existe un LED conectado al pin digital D13. Cuando el pin se encuentra en nivel alto, el LED está encendido, cuando el pin está a nivel bajo, el LED estará apagado.

El Arduino Nano V3 posee 8 entradas analógicas, cada una de ellas provee de 10 bits de resolución (1024 valores diferentes). Por defecto miden entre 5 voltios y masa, sin embargo es posible cambiar el rango superior usando la función `analogReference()`. También, algunos de estos pines poseen funciones especiales:

▪ **I2C**

Pines D4 (SDA) y D5 (SCL). Soportan comunicación I2C (TWI) usando la librería `Wire` (**60**).

- **AREF**

Tensión de referencia por las entradas analógicas. Se configura con la función `analogReference()`.

- **RESET**

Colocar esta línea a nivel bajo para resetear el microcontrolador. Se usa para añadir un botón de reset que mantiene a nivel alto el pin reset mientras no es pulsado.

3.1.2 DIGI XBee

El módulo de comunicación Digi XBee transmite la información procesada por el “Guante Electrónico de Datos” hacia la computadora. Trabaja con el protocolo 802.15.4 (ZigBee) para redes de sensores de gran alcance en enlaces inalámbricos, este tema fue abordado en el capítulo 2 en lo relacionado a protocolo I2C.



Cap.3: Fig. 3 DIGI 1mw XBee.

FUENTE: (63).

La comunicación del XBee con Arduino se ejecuta vía serial a través del modo AT²⁷ o el modo API²⁸. Estos módulos son configurados desde el computador utilizando el programa X-CTU o también desde un microcontrolador (63).

²⁷ Modo de transmisión serial transparente, se asemeja a una transmisión a través del puerto serial. El módulo sustituye a la línea serial, de modo que todos los datos recibidos por el UART a través del pin DIN se encola para transmitir. Todos los datos recibidos se envían al Arduino por el pin DOUT. En este modo, se pasa al modo de comandos AT.

²⁸ Este modo se basa en protocolos mediante tramas, en el que la aplicación interacciona con el módulo utilizando estructuras predefinidas tanto para la entrada como para la salida. Permite obtener e indicar mucha más información, soportar comandos ZDO y tráfico de perfil público, así como conocer el remitente de los mensajes, estar informado del éxito/fallo de envíos, etc...

3.1.2.1 CARACTERÍSTICAS

- Buen Alcance: hasta 100m en línea vista para los módulos XBee y hasta 1.6 Km para los módulos XBee Pro.
- 9 entradas/salidas con entradas analógicas y digitales.
- Consumo de corriente: Bajo consumo <50mA, cuando están en modo sleep <10uA, cuando transmite 45 mA, cuando recibe 50mA.
- 65,000 direcciones para cada uno de los 16 canales disponibles. Se tienen muchos de estos dispositivos en una misma red.
- Fáciles de integrar.
- Interfaz serial.

3.1.2.2 DESIGNACIÓN DE PINES



Cap.3: Fig. 4 Designacion pines Módulo XBee.

FUENTE: (62).

Pin #	Nombre	Dirección	Descripción
1	VCC	-	Terminal de alimentación
2	DOUT	Output	Terminal salida UART
3	DIN/CONFIG	Input	Terminal entrada UART
4	D08	Output	Salida digital 8
5	RESET	Input	Terminal de rest
6	PWM0 /RSSI	Output	Salida PWM 0/RX Muestra la intensidad de la señal.
7	PWM1	Output	Salida 1 PWM
8	(RESERVADO)	-	No conectado
9	DTR/ SLEEP_RQ/D18	Input	Control sleep ó entada digital 8
10	GND	-	Tierra

11	AD4 / DIO4	Either	Entada analógica 4 ó Digital I/O 4
12	CTS / DIO7	Either	Digital I/O 7
13	ON / SLEEP	Output	Módulo indicador de status
14	VREF	Input	Voltaje de referencia para entradas A/D
15	Associate/ AD5 / DIO5	Either	Entada / salida digital 5
16	RTS / AD6 /DIO6	Either	Entrada analógica 5 ó I/O Digital 5
17	AD3 / DIO3	Either	Entrada analógica 3 ó I/O Digital 3
18	AD2 / DIO2	Either	Entrada analógica 2 ó I/O Digital 2
19	AD1 /DIO1	Either	Entrada analógica 1 ó I/O Digital 1
20	AD0 / DIO0	Either	Entrada analógica 0 ó I/O Digital 0

Cap.3: Tabla. 2 Designacion de pines Módulo XBee.

FUENTE: (62).

3.1.2.3 INTERFAZ SERIAL TRANSPARENTE DE OPERACIÓN XBEE

El modo de transmisión transparente fue usado en el presente proyecto. Existen otros modos de operación de XBee que no se usan para este proyecto.²⁹

Este modo de transmisión de información por defecto en el módulo XBee está destinado principalmente a la comunicación punto a punto y remplazar alguna conexión serial por cable. En este modo, todo lo que ingresa por el pin 3 (Data in), es guardado en el bufer de entrada y luego transmitido. También, todo lo que ingresa como paquete RF, es guardado en el buffer de salida y luego enviado por el pin 2(Data out). En el modo transparente los módulos están configurados con comandos AT, pero la operación API no es compatible con este modo.

El módulo receptor del mensaje envía un paquete al módulo de origen llamado ACK que indica que el mensaje se recibió correctamente.

3.1.3 XBee Explorer Dongle

Dado que los equipos modernos no poseen un puerto serial, para la transmisión y recepción de datos entre la computadora y el “Guante Electrónico de Datos” vía

²⁹. Olimex (MCI Electronics) 02-11-2011, http://www.olimex.cl/pdf/Wireless/ZigBee/XBee-Guia_Usuario.pdf.

inalámbrica, se hizo necesario la incorporación de un dispositivo que permita emular un puerto serial a través de un puerto USB.

La tarjeta XBee Explorer Dongle permite generar un puerto serial virtual por medio del chip FTDI proporcionando así una comunicación confiable entre el módulo XBee y la computadora sin la necesidad de un cable.



Cap.3: Fig. 5 XBee Explorer Dongle.

FUENTE: (50).

3.1.3.1 CARACTERÍSTICAS.

- Leds para monitorear la actividad XBee.
- Led TX y RX para monitorear la actividad entre el módulo XBee y el chip USB.
- Conectores para módulo XBee.
- Soporta XBee serie 1 y serie 2.
- Ofrece una salida de corriente hasta 500mA.
- Usado para cargar el firmware a los dispositivos XBee.

3.1.4 MICROCONVERSION USB-serial CP2102

Dispositivo convertidor de USB a RS-232, con miniconector USB tipo B para conexión al computador, está basado en el chip CP2102 de Silicon Labs. Proporciona múltiples velocidades de transferencia en serie y acceso a las señales de control con un conector de 10 pines. Usado para pruebas iniciales del proyecto sobre un protoboard.



Cap.3: Fig. 6 Microconversor USB-serial CP2102.

FUENTE: (50).

3.1.5 IMU 6 GRADOS DE LIBERTAD ADXL345/ITG3200

IMU de 6 grados de libertad ensamblada por Sparkfun; posee una combinación de un acelerómetro y un giroscopio. El acelerómetro detecta la tasa de aceleración del guante, mientras que el giroscopio detecta los cambios rotacionales tales como el cabeceo, alabeo y guiñada.



Cap.3: Fig. 7 IMU 6 grados de libertad.

FUENTE: (50).

Los sensores se comunican por medio de I2C, cada sensor tiene una salida marcada, en el giroscopio INT0 y el acelerómetro con INT1, con alimentación de 3.3V.

Es necesaria la integración en una sola placa de estos dos sensores ya que en su combinación se obtienen datos mejor filtrados que al trabajarlos por separado. Sin embargo a continuación se detalla cada sensor de forma individual, para reconocer sus diferencias y el aporte significativo que cada uno brinda para la obtención de la información de la posición del guante en el espacio tridimensional.

3.1.5.1 ACELERÓMETRO ADXL345

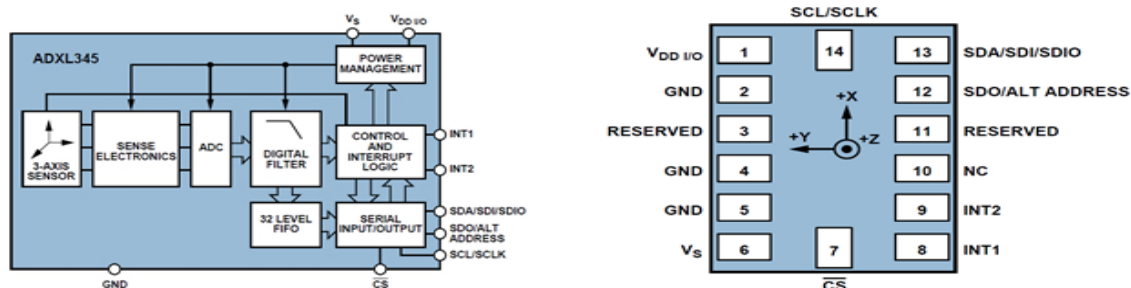
El ADXL345 es parte de la IMU de 6 grados de libertad, de tecnología MEMS, formado por una estructura micromaquinada suspendida sobre una oblea de silicio policristalino y capacitores diferenciales. Un plato de cada capacitor es conectado a

la estructura suspendida de forma independiente. Así la desviación de la estructura cambia la distancia entre los platos del capacitor, modificando la capacitancia. Un circuito en el interior del chip mide la diferencia entre estas capacitancias y el resultado es una amplitud proporcional de la aceleración. Esta amplitud es convertida en una señal digital compatible con el protocolo I2C (50).

3.1.5.1.1 CARACTERÍSTICAS

El ADXL345 es un acelerómetro MEMS de 3 ejes (X, Y, Z) mide aceleración estática de gravedad en aplicaciones de sensado de movimiento y aceleración dinámica en movimientos o golpes con salida digital, con las siguientes características:

- Consumo de energía 23uA a 2.5V.
- Selección del rango de medición por el usuario 2g, 4g, 8g,16g.
- Selección de la resolución de la salida desde 10 bits - 16 bits.
- Sensibilidad del sensor de 3.9 mg/LSB.
- Pila FIFO de 32 registros.
- Voltaje de alimentación de 2.0V a 3.6V.
- Interfaces digitales SPI de 3 y 4 hilos, I2C.
- Temperatura de trabajo (-40°C a + 85 °C)
- Dimensiones de encapsulado 3mm x 5mm x 1mm
- Selección del ancho de banda por comandos seriales.



Cap.3: Fig. 8 Diagrama de bloques y designación de pines ADXL345.

FUENTE: (50).

Pin	Nombre	Descripción
1	VDD I/O	Alimentación de voltaje
2	GND	Pin para conexión a tierra
3	RESERVED	Debe estar conectado a Vs o desconectado
4	GND	Pin para conexión a tierra
5	GND	Pin para conexión a tierra
6	Vs	Pin para alimentación de voltaje
7	CS	Entrada digital
8	INT1	Salida de interrupción 1
9	INT2	Salida de interrupción 2
10	NC	Sin conexión interna
11	RESERVED	Debe estar conectado a Vs o desconectado
12	SDO/ALT ADDRESS	Salida de datos serial(SPI 4 hilos)/ dirección I2C
13	SDA/SDI/SDIO	Entrada de datos I2C, serial SPI 4 hilos
14	SCL/SCLK	Comunicación serial del reloj. SCL para I2C y SCLK para SPI

Cap.3: Tabla. 3 Función de pines del ADXL345.

FUENTE: (50).

3.1.5.2 GIROSCOPIO ITG-3200

El segundo dispositivo que forma parte de la IMU de 6 grados de libertad es el ITG-3200, fabricado por InvenSense con tecnología MEMS está compuesto de 3 giroscopios independientes en cada eje que detectan la velocidad de rotación de coordenada (X, Y y Z) del sistema en un instante. El efecto Coriolis en la rotación del sensor en cualquier sentido, produce una deflexión que es detectada por un valor capacitivo. La señal que resulta luego de aplicar un movimiento al sensor es amplificada, demodulada y filtrada produciendo un voltaje proporcional a la velocidad angular. Este voltaje es digitalizado utilizando un convertidor analógico digital de 16 bits, que se encarga de hacer un muestreo en cada eje (X, Y y Z).

3.1.5.2.1 CARACTERÍSTICAS

- Salida digital en (X, Y y Z) en un circuito integrado.
- Sensibilidad de 14 LSB_S por °/s.

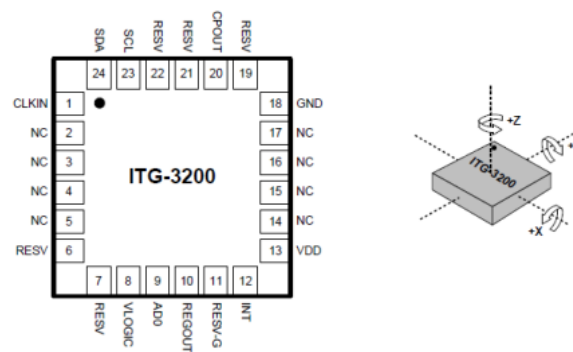
- Filtro programable digitalmente de paso bajo.
- Consumo de corriente de 6.5 mA (mayor duración de la batería).
- Amplio rango de suministro de voltaje de 2.1V a 3.6V.
- Sensor de temperatura con salida digital.
- Interface serial de modo rápido I2C.
- Entradas opcionales de reloj externo para sincronizar con el reloj del sistema.
- Presenta un rango completo de medición de +- 2000 grados cada segundo.

3.1.5.2.2 COMUNICACIÓN

El giroscopio ITG-3200 se comunica con el microcontrolador por medio de la interfaz I2C mediante la dirección 0x69(AD0 conectado en alto) o 0x68 (AD0 conectado en bajo). Luego del encendido el ITG-3200 se necesita 70 milisegundos para iniciar el funcionamiento (50ms a partir del encendido del giroscopio mas 20ms del registro de lectura y escritura de arranque). Hay que considerar que el microcontrolador conserva el último dato del giroscopio en sus registros.

Los datos de salida del sensor de temperatura y (X, Y y Z) como velocidades angulares, están disponibles a partir de los registros 0x1B A 0x22 que se encuentran almacenados como datos de 16 bits complemento a 2.

3.1.5.2.3 DIAGRAMA DE PINES ITG-3200



Cap.3: Fig. 9 Pines del giroscopio ITG-3200.

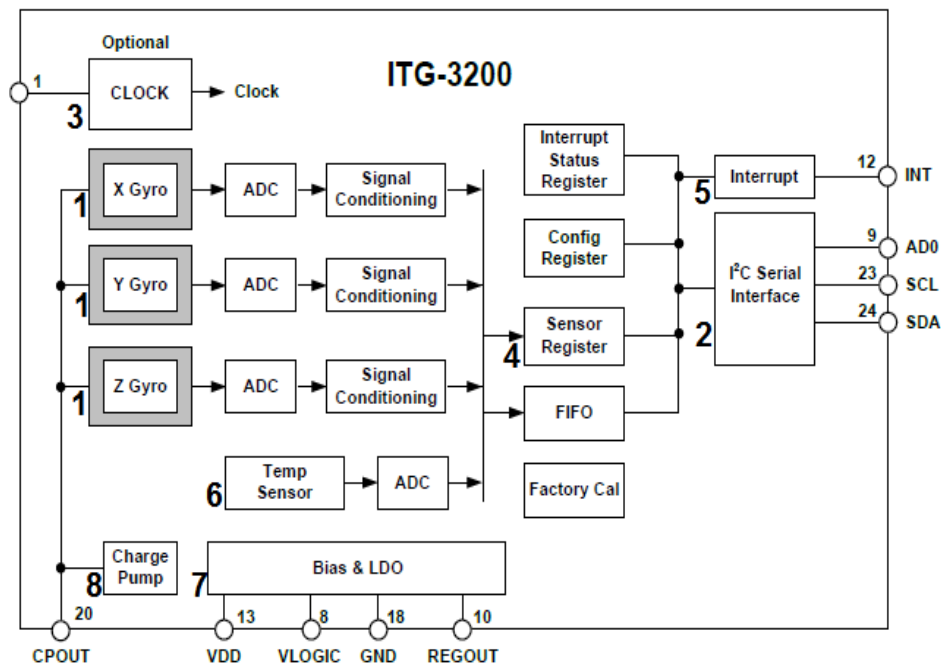
FUENTE: (50).

Número Pin	Nombre Pin	Descripción del Pin
1	CLKIN	Referencia opcional externa entrada de reloj. (Conecte a tierra si no se utiliza).
8	VLOGIC	Alimentación digital IO tensión de. VLOGIC debe ser \leq VDD en todo momento
9	AD0	I2c LSB
10	REGOUT	Conexión del capacitor regulador del filtro
12	INT	Interrupción de salida digital
13	VDD	Entrada de voltaje
18	GND	Entrada de suministro de tierra
11	RESV-G	Conexión a tierra (reservada)
20	CPOUT	Conexión al condensador
23	SCL	Reloj I2C
24	SDA	Datos I2C

Cap.3: Tabla. 4 Especificación de pines ITG-3200.

FUENTE: (50).

3.1.5.2.4 DIAGRAMA FUNCIONAL DE BLOQUES



Cap.3: Fig. 10 Diagrama funcional de bloques ITG-3200.

FUENTE: (50).

3.1.6 MPU-6050 (UNIDAD DE PROCESAMIENTO DEL MOVIMIENTO)

Para tomar la posición del “Guante electrónico de Datos” se usa el sensor inercial MPU-6050 de InvenSense. El principal motivo de la elección de este sensor es que; en un solo chip se integra un giroscopio y un acelerómetro que presenta una sobresaliente linealidad, reduciendo de esta manera significativamente el error de desalineación que se presenta cuando se fusionan los sensores dentro de una IMU.



Cap.3: Fig. 11 MPU6050.

FUENTE: (50).

La IMU de 6 grados de libertad vista anteriormente posee un acelerómetro ADXL345 y un giroscopio ITG-3200 por separado en un mismo tablero, en cambio la MPU-6050 es una unidad compacta para el procesamiento de movimiento elaborado con tecnología MotionProcessing³⁰ de Ivensense. Combina la aceleración con el movimiento de rotación y posee 6 grados de libertad. Está compuesto por un giroscopio de tres ejes, un acelerómetro de tres ejes y un Procesador de Movimiento Digital (DMP).

El DMP es un sistema basado en un microprocesador que posee un conjunto de instrucciones para el proceso de aplicaciones en tiempo real que requieran operaciones numéricas a muy alta velocidad, es utilizado para la salida de los ángulos de Euler y cuaterniones.

3.1.6.1 CARACTERÍSTICAS GENERALES

- Dimensiones de encapsulado 4x4x0.9mm.
- Salida digital de 6 ejes

³⁰ Tecnología registrada por IvenSense que combina una unidad inercial con el procesamiento digital y aplicaciones basadas en movimiento

- 16 bits de salida para los dos sensores
- Procesador DMP
- Selección del rango de medición por el usuario de 2g a 16g.
- Plataforma Android, Linux, Windows.
- Fuente de alimentación 2.37 V a 3.46V
- Comunicación I2C con todos los registros hasta 400 kHz en modo rápido.
- Filtros digitales para el acelerómetro, giroscopio y sensor de temperatura.
- Autodiagnóstico.

3.1.6.2 CARACTERÍSTICAS DEL GIROSCOPIO DEL MPU-6050

- Salida digital de rango de velocidad de ± 250 , ± 500 , ± 1000 , ± 2000 °/s
- Pin FSYNC para sincronización externa, se usa para la estabilización de imágenes en video y GPS.
- Bias mejorado y estabilidad en la sensibilidad de la temperatura que reduce la necesidad de la calibración por parte del usuario.
- Filtro digital paso bajo programable.
- Factor de escala predefinido por el fabricante.

3.1.6.3 CARACTERÍSTICAS ACELERÓMETRO DEL MPU-6050

- Salida digital de tres ejes con un rango de $\pm 2g$, $\pm 4g$, $\pm 8g$, $\pm 16g$.
- Salida de 16 bits.
- Funciona normalmente a 500 uA.
- A bajo voltaje funciona del siguiente modo:
 - 10 uA a 1.25 Hz.
 - 20 uA a 5Hz.

110 uA a 40Hz.

- Interrupciones programables por el usuario.
- Interrupción de caída libre.
- Interrupción a alta gravedad.
- Interrupción de cero movimientos.

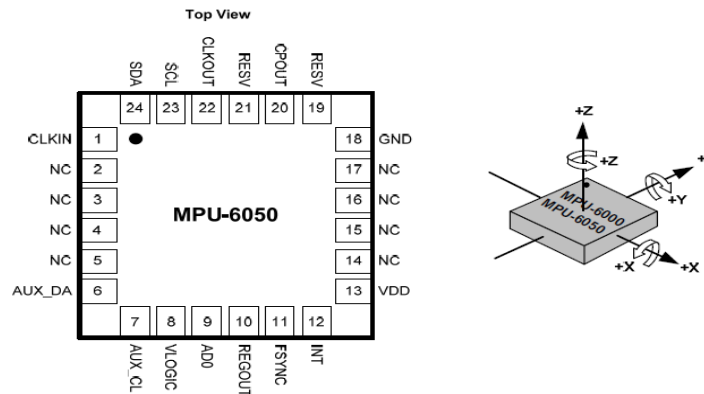
La MPU 6050 recoge los datos de los tres ejes del giroscopio, los tres ejes del acelerómetro y los datos de la temperatura; la tasa de muestreo para la sincronización de los datos es definida por el usuario.

3.1.6.4 DESIGNACIÓN DE PINES MPU-6050.

PIN	NOMBRE	DESCRIPCIÓN
1	CLKIN	Referencia opcional externa entrada de reloj. (Conecte a tierra si no se utiliza).
6	AUX_DA	I2C datos, para la conexión de sensores externos
7	AUX_CL	I2C reloj, para la conexión de sensores externos
8	VLOGIC	Entrada o salida digital de suministro de voltaje
9	AD0 / SDO	Dirección I2C, cuando el dispositivo trabaja como esclavo
10	REGOUT	Conexión al condensador regulador del filtro
11	FSYNC	Marco de entrada de sincronización digital. (Conecte a tierra si no se utiliza).
12	INT	Interrupción de salida digital (drenaje abierto)
13	VDD	Tensión de alimentación E / S digital
18	GND	Tierra.
20	CPOUT	Conexión al capacitor que anula el alto voltaje generado por la bomba de carga.
22	CLKOUT	Sistema de salida de reloj.
23	SCL	Señal de reloj I2C.
24	SDA	Línea de datos I2C.

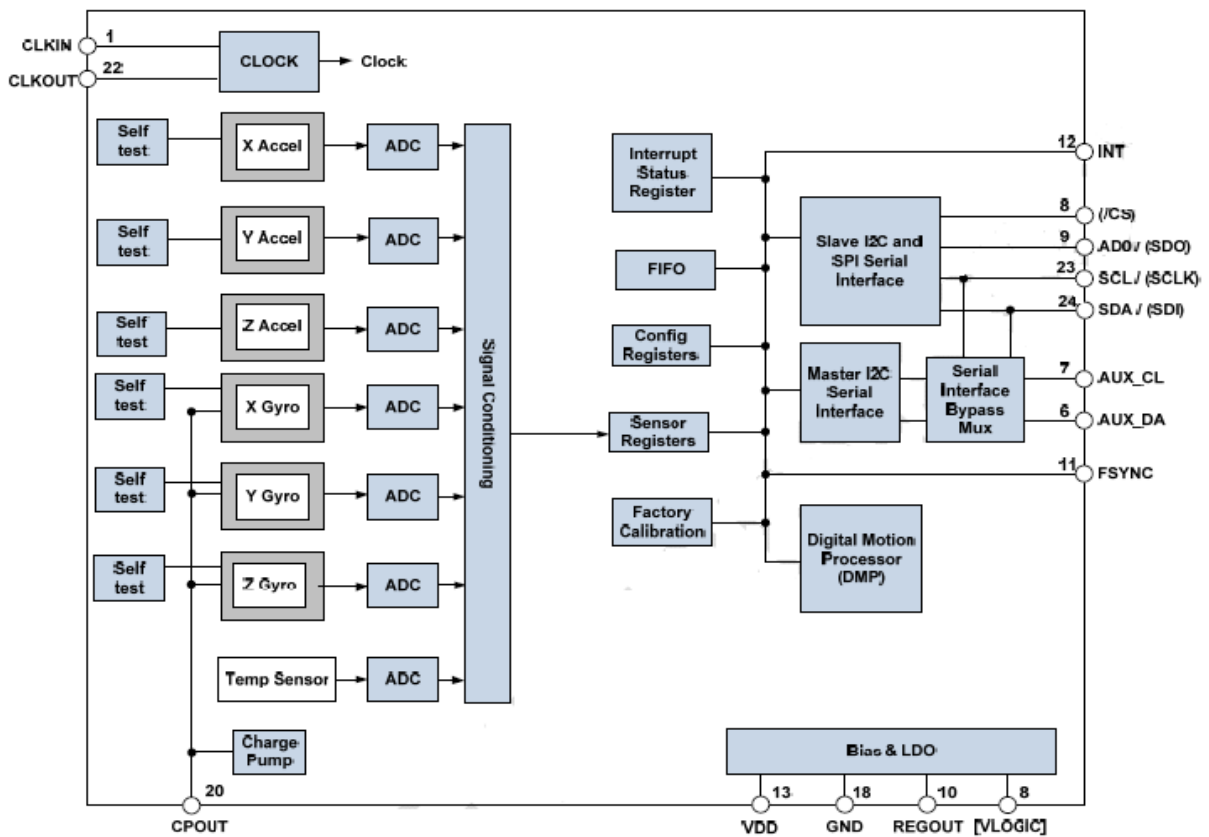
Cap.3: Tabla. 5 Designación de pines MPU-6050.

FUENTE: (64).



Cap.3: Fig. 12 Distribución de pines y polarización MPU-6050.
FUENTE: (64).

3.1.6.5 DIAGRAMA DE BLOQUES MPU-6050



Cap.3: Fig. 13 Diagrama de bloques MPU-6050.
FUENTE: (64).

La MPU-6050 se compone de las siguientes funciones y bloques:

▪ TRES BLOQUES PARA EL GIROSCOPIO MEMS CON 16 BITS ADC.

Los tres ejes del giroscopio detectan la rotación alrededor de los ejes X, Y, y Z. Cuando al giroscopio se lo hace girar alrededor de cualquier eje, el efecto Coriolis genera una vibración que es detectada por un pickoff³¹ capacitivo. La señal resultante es amplificada, demodulada y se filtra para producir una tensión proporcional a la velocidad angular. La frecuencia de muestreo del ADC es programable de 3,9 a 8000 muestras por segundo, y además el usuario puede seleccionar filtros paso bajos que permiten una amplia gama de frecuencias de corte.

▪ TRES BLOQUES PARA EL ACELERÓMETRO MEMS CON 16 BITS ADC.

Usa masas separadas para cada eje, la aceleración a lo largo de un eje en particular induce al desplazamiento de la masa correspondiente a cada eje, y los sensores capacitivos detectan el desplazamiento diferencial. Cuando el dispositivo se coloca sobre una superficie plana, en el eje X y Y se mide 0g mientras que en el eje Z se mide 1g. Cada sensor tiene un ADC para proporcionar salidas digitales.

▪ PROCESADOR DIGITAL DE MOVIMIENTO (DMP).

El DMP adquiere los datos del acelerómetro y giroscopio y los lee por medio de sus registros. El DMP tiene acceso a los pines exteriores de la MPU-6050 que se utilizan para generar interrupciones. Es usado como herramienta para minimizar la potencia, simplificar el tiempo y la arquitectura del software.

Por lo general los algoritmos de posicionamiento de movimiento se ejecutan a un ritmo de 200Hz con el fin de proporcionar resultados precisos con baja latencia.³²

³¹ Pickoff ó Captor capacitivo, Es un sistema que convierte el movimiento mecánico en una señal eléctrica proporcional, en el caso del giroscopio determina la posición del gimbal con respecto al giroscopio.

³² Tiempo que tarda un dato en estar disponible desde que se realiza su petición.

▪ **COMUNICACIÓN PRINCIPAL I2C.**

La MPU-6050 se comunica con el procesador del sistema actuando como esclavo utilizando el bus serial I2C. El nivel lógico para la comunicación con el maestro esta fijado por la tensión en VLOGIC.

▪ **INTERFAZ AUXILIAR I2C**

La MPU-6050 posee un bus auxiliar para la comunicación con el procesador del sistema a través del bus I2C con un magnetómetro de tres ejes u otros sensores.

▪ **RELOJ.**

La MPU-6050 tiene un sistema flexible de reloj tanto para fuentes de reloj externas o internas. La exactitud del reloj es importante, ya que los errores de tiempo afectan directamente a los cálculos de distancia y el ángulo que ejecuta el DMP.

▪ **REGISTRO DE DATOS DEL SENSOR**

Contiene los últimos datos de medición del giroscopio, acelerómetro, sensor auxiliar y temperatura. Los registros son de sólo lectura y son leídos en cualquier momento. Sin embargo, la función de interrupción suele ser utilizada para determinar cuando los nuevos datos se encuentran disponibles.

▪ **FIFO**

Registro de configuración de 1024 bytes determina los datos del giroscopio, del acelerómetro, las lecturas de temperatura y los auxiliares de entrada. Un contador FIFO realiza un seguimiento de cuántos bytes de datos válidos se encuentra en el FIFO.

▪ **AUTO TEST DEL GIROSCOPIO Y ACELERÓMETRO.**

El autodiagnóstico permite probar las porciones mecánicas y eléctricas de los sensores, tomando para cada eje una medición por medio del control de los bits del giroscopio y el control de registros del acelerómetro. Cuando el autodiagnóstico es activado los sensores producen una señal de salida.

▪ **INTERRUPCIONES**

El sistema de interrupción se utiliza en el generador de reloj para especificar si los nuevos datos están disponibles para ser leídos. Los datos son leídos desde el registro de estado de interrupción. La interrupción se configura a través del registro de configuración de interrupciones por medio del pin INT.

Los motivos para producir una interrupción son:

- Cuando los datos están disponibles para ser leídos
- Cuando el MPU-6050 no recibe un reconocimiento de un sensor auxiliar por el bus I2C auxiliar.

▪ **SENSOR DIGITAL DE TEMPERATURA**

Sensor de temperatura digital, es utilizado para medir la temperatura actual del MPU-6050.

▪ **BIAS Y LDO**

El BIAS y LDO, toman una tensión no regulada del VDD y generan una fuente interna estable, así como las corrientes requeridas por el MPU-6050. Sus dos entradas son VDD no regulado de 2.37 a 3.46V y una referencia VLOGIC de una alimentación de tensión de 1.7 V.

▪ **BOMBA DE CARGA**

La Bomba de carga genera alta tensión necesaria para alimentar los osciladores del MEMS. Su salida es anulada en el capacitor conectado al pin CPOUT.

3.1.6.6 INTERRUPCIONES MPU-6050

La MPU-6050 posee un sistema de interrupciones programable mediante la señal de interrupción en el pin INT. La fuente de la interrupción se muestra mediante banderas de estado y se activan y desactivan de forma individual.

INTERRUPCIÓN	MÓDULO
FREE FALL DETECTION	FREE FALL
MOTION DETECTION	MOTION
ZERO MOTION DETECTION	ZERO MOTION
FIFO OVERFLOW	FIFO
DATA READY	SENSOR REGISTERS
I2C MASTER ERROR	I2C MASTER
I2C SLAVE	I2C MASTER

Cap.3: Tabla. 6 Interrupciones MPU-6050.

FUENTE: (64).

▪ **INTERRUPCIÓN FREE FALL**

La interrupción Free Fall (caída libre) se detecta comprobando si las mediciones de los 3 ejes del acelerómetro tienen un valor absoluto por debajo del umbral de aceleración programado por el usuario.

Por cada muestra donde se cumple la condición del usuario se incrementa un contador, cuando no se cumple la condición, el contador se decrementa. Una vez que el contador llegue a un umbral programable por el usuario (contador de umbral), la interrupción de caída libre se dispara y se activa una bandera. La bandera se borra una vez que el contador se decrementa a cero. El contador no se incrementa por encima del valor de umbral del contador o disminuye por debajo de cero.

Existen varios parámetros de configuración para detectar con precisión la detección de caída libre. El umbral de aceleración y el umbral del contador son configurables por el usuario. El registro FF_THR permite al usuario establecer un umbral en incrementos de 1mg. El registro FF_DUR permite al usuario establecer la duración en incrementos de 1ms.

▪ **MOTION INTERRUPT**

La MPU 6050 proporciona la capacidad de detección de movimiento con una funcionalidad similar a la detección de caída libre. La mediciones del acelerómetro

pasa a través de un filtro tipo paso alto configurable (DHPF), con el fin de eliminar el bias debido a la gravedad.

El registro MOT_THR establece un umbral en incrementos de 1mg. El registro MOT_DUR especifica la duración en incrementos de 1ms.

▪ **ZERO MOTION INTERRUPT**

Para la detección de cero movimiento se utiliza el filtro digital tipo paso alto (DHPF), este esquema es similar al umbral de detección de caída libre (free fall). Cada eje de la medición del acelerómetro de filtro paso alto debe tener un valor absoluto menor que un umbral especificado en el registro ZRMOT_THR, que se incrementa en 1mg. Cada vez que se cumple esta condición, se incrementa un contador; cuando este contador alcanza un umbral especificado en ZRMOT_DUR se genera una interrupción.

A diferencia de Free Fall o Motion Detection, Zero Motion (cero movimiento) provoca una interrupción de movimiento tanto en el momento que se detecta cero movimiento como cuando ya se detecta el movimiento.

3.1.6.7 INTERFAZ I2C DE MPU-6050

I2C es una interfaz de dos hilos compuesta por las señales de datos SDA y el reloj SCL visto en el capítulo 2. La MPU-6050 siempre funciona como un dispositivo esclavo cuando se comunica con el procesador del sistema como maestro. Las líneas SDA y SCL trabajan con resistencias pull-up a VDD. La velocidad máxima del bus es de 400 kHz.

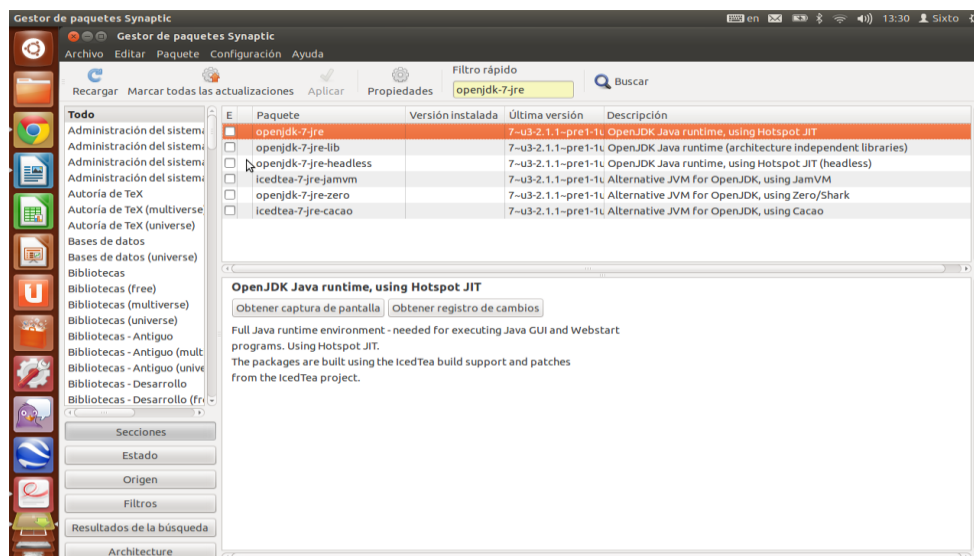
3.2 SOFTWARE

3.2.1 INSTALACIÓN DE ARDUINO IDE EN UBUNTU LINUX

Este procedimiento es más sencillo si se realiza la instalación de este software desde los repositorios del sistema operativo Ubuntu por medio de su Centro de

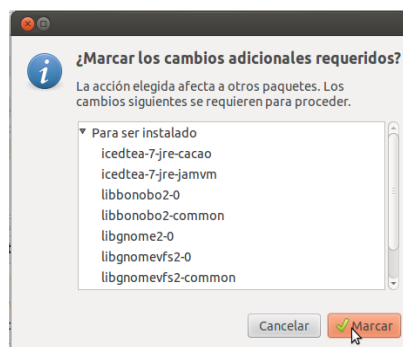
Software; sin embargo si se desea obtener una versión actualizada del IDE, con todas sus funcionalidades se debe efectuar el siguiente procedimiento:

- El primer paso consiste en abrir el gestor de paquetes Synaptic de Ubuntu, se escribe “*openjdk-7-jre*”, en campo de texto “*filtro rápido*”, se selecciona el paquete (Fig.14), luego se marca haciendo click en el botón respectivo (Fig.15), para una posterior instalación. Esta acción permite instalar el jdk necesario para la ejecución de Arduino IDE.



Cap.3: Fig. 14 Selección del paquete *openjdk-7-jre* y sus dependencias

FUENTE: AUTOR.

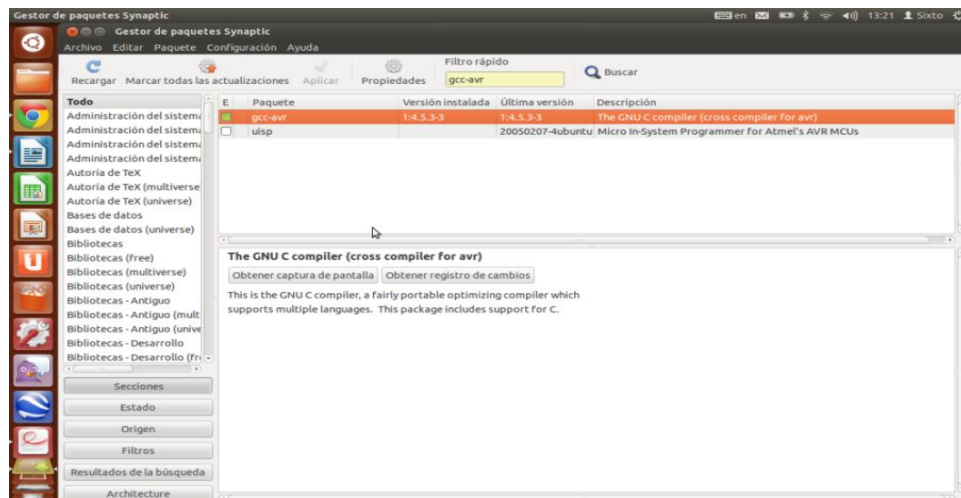


Cap.3: Fig. 15 Dependencias del paquete *openjdk-7-jre*.

FUENTE: AUTOR.

- En el siguiente paso se procede a instalar el compilador de C el “GCC-AVR” que tiene el código de lenguaje de alto nivel y crea un binario, el cual es utilizado

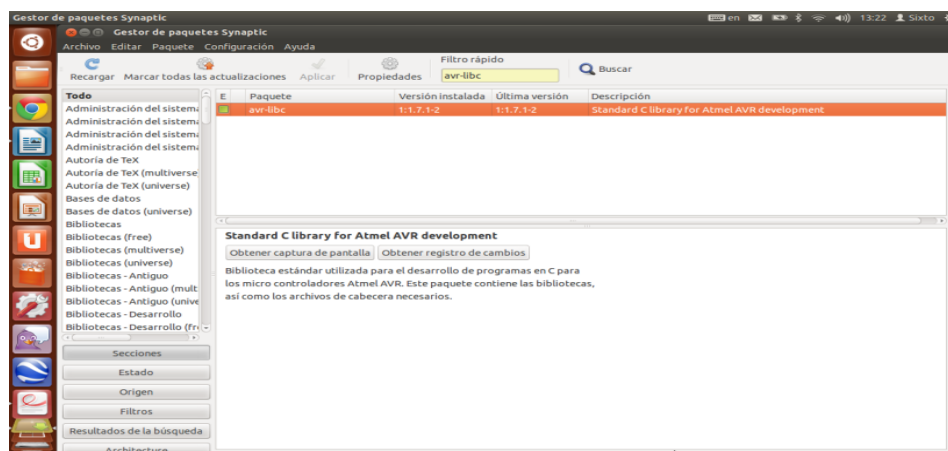
para programar a los microcontroladores AVR. Este cometido se consigue escribiendo “gcc-avr” en el campo de texto mencionado en el paso 1 y haciendo la selección respectiva como sugiere Fig.16.



Cap.3: Fig. 16 Selección del compilador gcc-avr.

FUENTE: AUTOR.

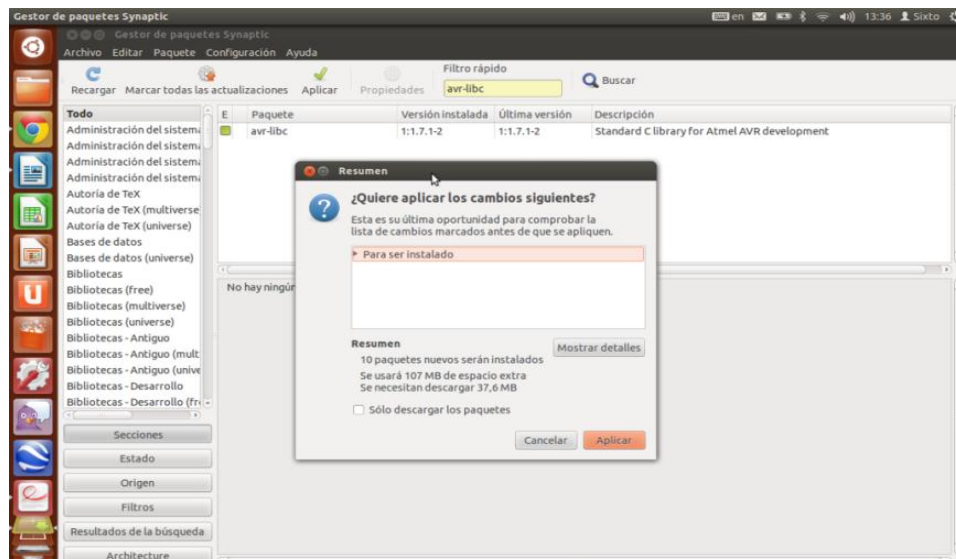
- En este punto se instala “avr-libc” que es un conjunto de bibliotecas de alta calidad basadas en C para su uso con “AVR- GCC” en los microcontroladores AVR Atmel. El proceso de selección es similar a los dos casos anteriores y se indica en la captura de Fig.17.



Cap.3: Fig. 17 Selección de las librerías “avr-libc”.

FUENTE: AUTOR.

- Una vez completado los pasos anteriores se procede a instalar los paquetes seleccionados, la descarga de Internet e instalación lo gestiona el sistema operativo de forma automática, ver Fig.18.



Cap.3: Fig. 18 Descarga e instalación de paquetes seleccionados en Ubuntu 12.04.

FUENTE: AUTOR.

- Un último paso y el más importante consiste en descargar Arduino IDE en su versión más reciente desde su página principal, ubicarlo en el disco y ejecutarlo para poder editar los Sketch **(65)**.

3.2.2 PUERTO SERIAL EN LINUX

Para Linux los periféricos se representan como ficheros del sistema de archivos. De esta manera cuando se envían datos a través del puerto serial, se abre el fichero asociado al puerto y se escribe con las funciones ya conocidas. Para cada puerto serial en Linux existe uno o más ficheros de dispositivos. Cabe mencionar que la notación `/dev/ttySx` hace referencia a los puertos nativos de la PC, para puertos emulados, el sistema operativo Linux usará la notación `/dev/ttyUSBx`, como se muestra en la tabla 7.

PUERTO	FICHERO
COM1	/dev/ttyUSB0
COM2	/dev/ttyUSB1
COM3	/dev/ttyUSB2
COM4	/dev/ttyUSB3

Cap.3: Tabla. 7 Asociación de puertos serial a ficheros de dispositivo.

FUENTE: (66).

3.2.2.1 FUNCIONES QUE PERMITEN INTERACTUAR CON EL PUERTO SERIAL

▪ FUNCIÓN OPEN

Para acceder al puerto serie se hace una llamada a la función Open y se indica el fichero que se desea abrir (“/dev/ttyUSBx”) y las opciones de apertura necesarias. Dichas opciones consisten en constantes predefinidas. Si el fichero se abre correctamente, la función retorna un descriptor de archivo (fd) cuyo valor es un entero positivo que se utilizará posteriormente en operaciones de entrada y salida. Mediante las funciones read y write. Si el fichero no se abre la función retornará un número negativo.

```
int open(const char * nombrePuerto, int opciones);
```

▪ FUNCIÓN WRITE

Para escribir datos en el puerto serial se hace una llamada a la función write. Esta función envía los datos al puerto, escribe un número de bytes en el fichero referenciado por fd. Para obtener más información acerca de ésta y otras funciones relacionadas se escribe en una consola la orden *man 2 write*.

```
ssize_t write(int fd, const char * dato, size_t num);
```

▪ FUNCIÓN READ

La función read lee un número específico de bytes del fichero fd y los guarda en una zona de memoria, su estructura es la siguiente:

```
ssize_t read(int fd, char * datos, size_t numBytes);
```

3.2.2.2 ENTRADA PARA DISPOSITIVOS SERIALES

▪ MODO DE ENTRADA CANÓNICO

Proceso normal para terminales y comunicación con otros dispositivos. La entrada se procesa en unidades de línea, quiere decir que un read devolverá una línea completa de entrada. Este proceso maneja también los caracteres borrado, borrado de palabra, reimprimir caracter, traducir CR (retorno de carro fin de línea en DOS), NL (nueva línea), etc.

▪ MODO DE ENTRADA NO CANÓNICO

Manipula un número fijo de caracteres por lectura y permite un carácter de temporización. Este modo se usa cuando la aplicación lee un número fijo de caracteres o también si el dispositivo envía ráfagas de caracteres. Dado que el guante electrónico de datos envía constantemente ráfagas de caracteres, este modo es el adecuado y finalmente implementado para la recepción de datos en Ubuntu Linux.

▪ ENTRADA SINCRÓNICA Y ASINCRÓNICA

Los procesos canónico y no canónico se utilizan de dos maneras.

Modo sincrónico. La sentencia read se bloquea hasta que la lectura esté completa.

Modo asincrónico. La sentencia read devuelve inmediatamente y envía una señal al programa llamador a través de un manejador de señales cuando la señal esté completa.

3.2.2.3 CONFIGURACIÓN DEL PUERTO SERIAL EN LINUX.

Todos los parámetros se configuran fácilmente a través de la interfaz de POSIX(Portable Operating System Interface) con un programa basado en C, la

configuración se guarda en una estructura “*struct termios*”, que está definida en el archivo “*termios.h*”.

MIEMBRO	DESCRIPCIÓN
<code>c_cflag</code>	Opciones de control
<code>c_lflag</code>	Opciones locales
<code>c_iflag</code>	Opciones de entrada
<code>c_oflag</code>	Opciones de salida
<code>c_cc</code>	Caracteres de control
<code>c_ispeed</code>	Velocidad (baudios) de entrada
<code>c_ospeed</code>	Velocidad (baudios) de salida

Cap.3: Tabla. 8 Miembros de la estructura *termios*.

FUENTE: (66).

Las funciones para la configuración del puerto serial son:

tcgetattr: Obtiene los parámetros actuales del puerto asociados con el objeto referido por `fd` y los guarda en la estructura *termios* referenciada por `termiosPuerto`.

```
int tcgetattr(int fd, struct termios * termiosPuerto);
```

tcsetattr: Se utiliza para actualizar los parámetros del puerto representado por el descriptor de archivo `fd`, desde la estructura *termios* referenciada por `termiosPuerto`.

```
int tcsetattr(int fd, struct * termiosPuerto);
```

tcflush: Se utiliza para limpiar los buffers de entrada y salida.

```
int tcflush ( int fd, int selector );
```

La variable `selector` esta representada por uno de tres valores diferentes y dependiendo de estos su funcionalidad varía de forma sustancial.

TCIFLUSH. Desecha datos recibidos pero no leídos.

TCOFLUSH. Desecha datos escritos pero no transmitidos.

TCIOFLUSH. Desecha datos recibidos pero no leídos y los escritos pero no transmitidos.

cfsetospeed: Establece la velocidad de salida que está guardada en la estructura `termios` apuntada por `termiosPuerto`.

```
int cfsetospeed ( struct termios * termiosPuerto, speed_t velocidad);
```

cfsetispeed: Establece la velocidad de entrada guardada en la estructura `termios` en `termiosPuerto`,

```
int cfsetispeed ( struct termios * termiosPuerto, speed_t velocidad);
```

Para los dos casos los valores se hacen efecto cuando se llama con éxito a `tcsetattr()`.

3.2.2.4 OPCIONES DE CONTROL.

Para el control de la velocidad del puerto serial, el número de bits de datos, bits de parada, la paridad y el control del flujo de datos por hardware se utiliza la bandera `c_cflag` de la estructura `termios`.

▪ OPCIONES LOCALES

Para las opciones locales, la bandera `c_lflag` permite controlar la forma en que serán tratados los caracteres de entrada que puede ser canónica o no canónica.

▪ OPCIONES DE ENTRADA

El miembro `c_iflag` permite controlar el procesamiento de caracteres de entrada que se reciben por el puerto.

▪ OPCIONES DE SALIDA

La bandera `c_oflag` contiene las opciones para el procesamiento de la salida de datos.

▪ CARACTERES DE CONTROL

Por medio del array de caracteres `c_cc` se configura los caracteres de control y los parámetros de los timeout o tiempos de espera en la comunicación serial.

3.2.2.5 CÓDIGO PARA PARA LA COMUNICACIÓN CON EL GUANTE ELECTRÓNICO DE DATOS.

Para acceder al puerto serial en Ubuntu Linux se implementó la clase *“Puerto”* cuya gestión consiste en efectuar las operaciones de configuración, apertura, lectura, escritura y cierre del puerto. En el archivo `puerto.h` se encuentran las especificaciones de las funciones que desempeñan estos roles y en el archivo `puerto.cpp` la respectiva implementación de las mismas. Para ver la implementación completa con sus respectivos comentarios de esta clase refiérase al Anexo A.

3.2.3 INTERACCIÓN GUANTE DE DATOS - PC

El guante electrónico de datos envía de forma inalámbrica y permanente tres valores, dos de los cuales representan las coordenadas X , Y para el movimiento del puntero del mouse y un último que identifica el accionamiento de cualquiera de los ocho sensores de contacto representado por un código específico.

Los datos que recibe el computador a través del puerto serial deben ser interpretados y traducidos para generar eventos propios de un mouse aéreo y de forma opcional también se pueden elegir ciertos eventos de teclado.

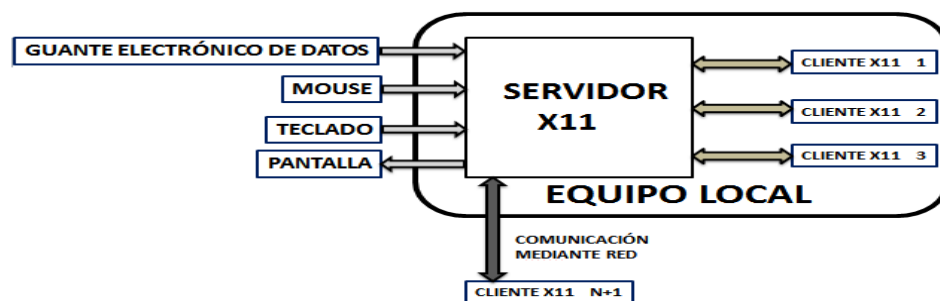
Para hacer posible la ejecución de los eventos fue necesaria la utilización de las bibliotecas `XLib` y `XTest` del sistema `X11` las que contienen el software necesario para acceder a los drivers del mouse y del teclado del sistema operativo Ubuntu.

3.2.3.1 SISTEMA X11

El sistema `X11` o también conocido como `X` ó `XWindows System`, es un sistema de ventanas basado en la arquitectura cliente servidor desarrollado en los años 80s por el MIT, este sistema se encarga de gestionar el entorno gráfico en los sistemas UNIX

y sus derivados permitiendo la interacción gráfica en red entre un usuario y una o varias computadoras. El X11 se ejecuta en el servidor y se encarga de controlar la pantalla a través de la entrada de los datos del teclado, mouse o del “Guante Electrónico de Datos” como es el caso de este proyecto.

X11 como servidor trabaja a través del monitor, teclado, mouse y el “Guante Electrónico de Datos”; y sus clientes son las aplicaciones o programas representados en cada ventana.



Cap.3: Fig. 19 Arquitectura X11.

FUENTE: AUTOR.

X11 define tres componentes: el servidor, los programas clientes y el canal de comunicaciones.

Para manejar las entradas y salidas de los dispositivos de hardware X11 realiza lo siguiente: despliega los gráficos en la pantalla, recibe las entradas del teclado y guía el cursor según el movimiento del mouse o el “Guante Electrónico de Datos”.

Las tareas básicas de X11 son:

- Permitir el acceso a los dispositivos para varios clientes.
- Mantenimiento de las estructuras de datos necesarios en la ejecución de los programas.
- Interpretar los mensajes de la red para los clientes.

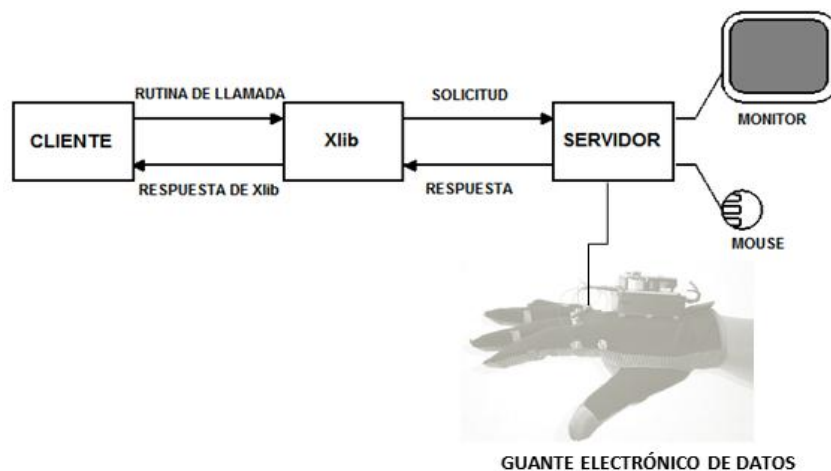
3.2.3.1.1 BIBLIOTECA XLIB

La comunicación entre el cliente y el servidor en X11 se logra a través del protocolo Xprotocol, que es un conjunto de comandos básicos con los que se puede generar

ventanas, posicionarlas y controlar sus eventos. Xprotocol intercambia datos de forma asincrónica por una vía de comunicación bidireccional que permite la transmisión de una secuencia de bytes.

Para acceder al protocolo Xprotocol, X11 hace uso de la biblioteca Xlib. Ésta biblioteca contiene funciones en lenguaje C que son usadas por aplicaciones o programas cliente para comunicarse con el sistema de ventanas. Las funciones de Xlib manejan tareas como por ejemplo: abrir una pantalla, manipular ventanas, dibujar y manejar eventos.

Las llamadas hechas al cliente se envían por medio de la biblioteca Xlib y la información que devuelve el cliente pasa nuevamente por la biblioteca Xlib convirtiendo la información en un lenguaje X11 estándar.



Cap.3: Fig. 20 Relación Cliente-XLib-Servidor.

FUENTE: (67).

En Fig.20 se presenta las relaciones entre el cliente, XLib y el servidor. El cliente llama a las rutinas de XLib, las que siempre residen en el cliente. XLib internamente llama y devuelve la información al cliente cuando sea apropiado. Cuando una XLib requiere la intervención del servidor genera y envía una solicitud al servidor.

XLib permite a un cliente comunicarse con el servidor para crear y administrar:

- Las conexiones entre el servidor y el cliente.
- Ventanas.

- Colores.
- Imágenes (ancho y estilo de línea).
- Cursores.
- Fuentes de texto.
- Envío de gráfico entre clientes.
- Gestión de eventos.
- Gestión de errores.

3.2.3.1.2 EVENTOS DE XLIB

Los eventos en XLib son mensajes que se envían desde el servidor al cliente para que este sea informado de lo que ocurre en el display del servidor. Los eventos son provocados por hardware cuando: se presiona el botón del mouse o el Guante Electrónico de Datos, se mueve el mouse, se presiona una tecla, mientras que por software sucede un evento cuando: se cambia el tamaño de la ventana, se coloca el foco sobre una ventana, se ordena las ventanas apiladas, etc.

	EVENTO
RATÓN O GUANTE ELECTRÓNICO DE DATOS	ButtonPress, ButtonRelease, MotionNotify, EnterNotify, LeaveNotify, MappingNotify
TECLADO	KeyPress, KeyRelease, FocusIn, FocusOut, KeymapNotify, MappingNotify
VENTANAS	ConfigureNotify, CreateNotify, DestroyNotify, MapNotify, UnmapNotify, GravityNotify, VisibilityNotify, ReparentNotify
GRÁFICOS	Expose, GraphicsExpose, NoExpose, ColormapNotify
ADMINISTRADOR DE VENTANAS	CirculateRequest, ConfigureRequest, MapRequest, ResizeRequest
COMUNICACIÓN ENTRE CLIENTES	ClientMessage, PropertyNotify, SelectionClear, SelectionRequest, SelectionNotify

Cap.3: Tabla. 9 Eventos Xlib.

FUENTE: (68).

3.2.3.2 CONEXIÓN CON EL SERVIDOR X11

Antes de que un programa cliente pueda usar la pantalla o “*Display*”, se debe primero establecer una conexión con el servidor X, una vez establecida la conexión, se utilizan las macros y funciones de XLib con diferentes propósitos.

Para abrir una conexión con el servidor X que controla la pantalla se utiliza la función “*XOpenDisplay*” cuyas características se explican a continuación:

Sintaxis

```
Display * XOpenDisplay(char * nombreDisplay);
```

Argumentos

“*nombreDisplay*”. Especifica el nombre del “*Display*” (hardware) que determina el “*Display*” propiamente dicho y el dominio de comunicación a utilizar. En un sistema POSIX, por defecto la variable “*nombreDisplay*” toma el valor de NULL.

Valor de Retorno

Si la conexión tiene éxito, la función “*XOpenDisplay*” devuelve una estructura “*Display*” definida en “*X11/Xlib.h*” que contiene toda la información acerca del servidor X, si no tiene éxito, devuelve NULL.

El valor devuelto por “*XOpenDisplay*” debe ser pasado como argumento a la función “*DefaultRootWindow*” la cual retorna a su vez un objeto “*Window*” que representa la ventana actual o por defecto sobre la que se ejecutarán las operaciones con el servidor X a lo largo del programa cliente.

3.2.3.3 DESCONECTAR CON EL SERVIDOR X11

La función “*XCloseDisplay*” cierra la conexión con el servidor X para el display especificado en la estructura “*Display*” y destruye todas las ventanas.

Sintaxis

XCloseDisplay (display)*

Argumentos

“*display*” especifica la conexión con el servidor X.

3.2.3.4 EXTENSIÓN XTest

XTest está compuesta por un conjunto mínimo de las extensiones cliente-servidor que se necesita para probar por completo X11 sin intervención del usuario; ofreciendo un grupo mínimo de recursos que solucionan problemas inmediatos de prueba y validación **(69)**.

Los objetivos de XTest son:

- Minimizar los cambios necesarios en el resto del servidor.
- Minimizar los efectos negativos sobre el rendimiento durante la operación normal del servidor.
- Limitar la extensión a un nivel alto dentro del servidor, minimizando los problemas de portabilidad.

3.2.3.4.1 FUNCIONES XTest

▪ OPERACIONES DE CLIENTE

Las operaciones del cliente manipulan cierto comportamiento de parte del cliente que sin ellas quedaría oculto. Su implementación depende de los detalles del enlace del lenguaje real y del nivel de almacenamiento intermedio de petición.

Las operaciones de cliente para XTest son:

- XTestSetGContextOfGC.
- XTestSetGVisualIDOfVisual.
- XTestDiscard.

▪ PETICIONES DEL SERVIDOR

XTestGetVersion. Permite a un cliente especificar un número de versión mayor o menor al del servidor y a éste, que pueda responder con sus propias versiones.

XTestCompareCursor. Permite el acceso a un recurso del servidor.

XTestFakeInput. Permite la síntesis limitada de sucesos de los dispositivos de entrada.

3.2.3.4.2 FUNCIONES QUE PERMITEN SIMULAR EVENTOS

“XTestFakeMotionEvent”. Esta función hace una petición al servidor X para simular el movimiento del puntero en una posición específica de la ventana raíz *“root_window”*.

Sintaxis

*XTestFakeMotionEvent(Display * display, Window root_window, int x, int y, unsigned long delay)*

Argumentos

“display” Especifica la conexión con el servidor.

“root_window” Ventana raíz donde se manifiesta el evento.

“int x, int y” Coordenadas para el posicionamiento del puntero.

“delay” Tiempo en milisegundos que dura el evento.

“XTestFakeButtonEvent”. Esta función envía una petición al servidor para simular eventos del mouse, tales como clic izquierdo, doble clic, clic derecho y drag sobre la ventana raíz *“root_window”*.

Sintaxis

*XTestFakeButtonEvent(Display * display, unsigned int button, bool is_press, unsigned long delay)*

Argumentos

“display” Especifica la conexión con el servidor.

“*button*” Indica el botón a ser simulado, la variable “*button*” toma tres valores diferentes dependiendo del evento, así se tiene que para clic izquierdo tomará el valor de 1, botón central o scroll 2 y clic derecho 3.

“*is_press*” Toma dos valores posibles; 1 o true si un determinado botón es presionado y 0 o false en caso contrario.

“*delay*” Especifica el tiempo en milisegundos que dura el evento.

“*XTestFakeKeyEvent*” Genera una petición al servidor X para simular un “*KeyPress*”, esta función tiene la capacidad de generar un evento al ser pulsada cualquier tecla ya que el archivo de cabecera “*keysym.h*” contiene las constantes simbólicas necesarias para la mayoría de distribuciones de teclado y sus códigos respectivos.

Sintaxis

XTestFakeKeyEvent (*Display * display, unsigned int keycode, bool is_press, unsigned long delay*)

Argumentos

“*display*” Especifica la conexión con el servidor.

“*keycode*” Variable que contiene el código correspondiente de la tecla a ser simulada.

“*is_press*” Toma dos valores posibles; 1 o true si una tecla es presionada y 0 o false en caso contrario.

“*delay*” Especifica el tiempo en milisegundos que dura el evento.

3.2.3.5 INSTALACIÓN DEL API XLIB Y XTEST EN UBUNTU LINUX

El proceso de instalación de estas librerías es bastante sencillo en Ubuntu Linux, solo se debe escribir las siguientes líneas de comando en una consola.

Para la instalación de la librería XLib:

```
sudo apt-get install libx11-dev
```

Para la instalación del paquete XTest:

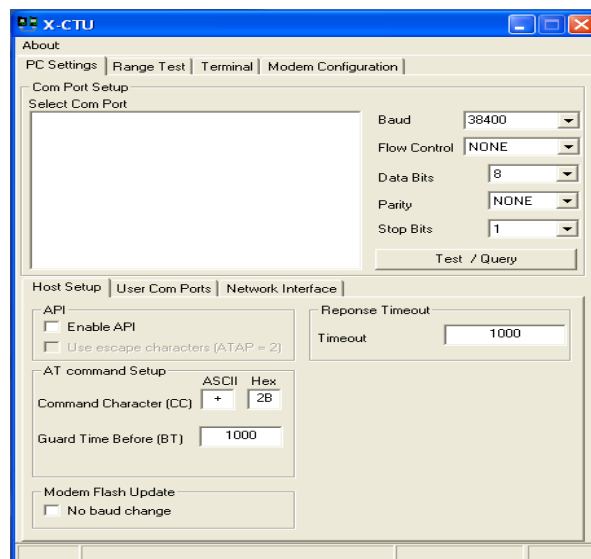
```
sudo apt-get install libxtst-dev
```

3.2.3.6 IMPLEMENTACIÓN DE UN PROGRAMA CLIENTE EN C++ PARA LA GESTIÓN DE EVENTOS

Con la finalidad generar un aplicativo que permita interpretar los movimientos del “Guante Electrónico de Datos”, tanto de su posición en el espacio como el accionamiento de sus sensores de contacto efectuados por los dedos del usuario; se desarrolló la clase XUTIL, declarada en el archivo xutil.h e implementada en el archivo xutil.cpp que se explica en forma detallada en el Anexo B.

3.2.4 X-CTU SOFTWARE

X-CTU es una aplicación basada en Windows que proporciona Digi internacional, es una herramienta bastante cómoda para la configuración de los módulos Xbee, con este software, el usuario será capaz de actualizar el firmware, actualizar los parámetros y efectuar pruebas de comunicación de una forma fácil e intuitiva. Esta aplicación se descarga gratuitamente desde la Web de Digi (70).



Cap.3: Fig. 21 Interfaz de usuario X-CTU.

FUENTE: AUTOR.

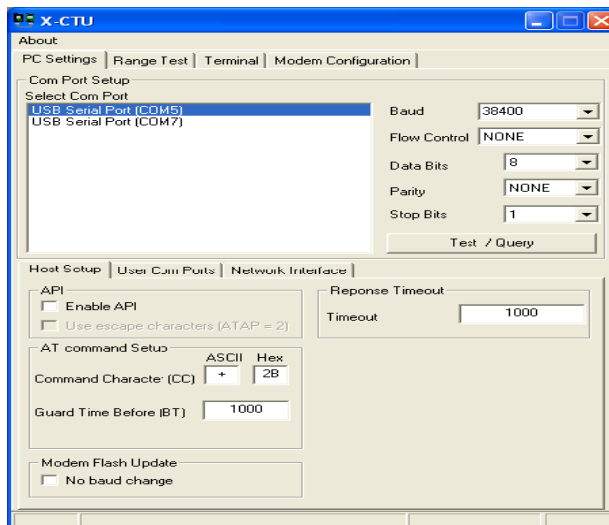
3.2.4.1 CARACTERÍSTICAS

El software X-CTU tiene cuatro pestañas principales que permiten configurar de una forma sencilla a los módulos Xbee. Una breve descripción de cada una de ellas se describe a continuación:

▪ **PC SETTINGS.**

Esta pestaña permite la selección y configuración del puerto serial a través de los diferentes parámetros que son:

- **Select Com Port:** En este cuadro de texto se enlista todos los puertos seriales disponibles y permite la selección del puerto con el cual se efectuará la configuración del módulo Xbee, ver Fig.22.
- **Baud:** En esta lista se selecciona la velocidad de las señales por segundo con la cual se va a transmitir.
- **Flow Control:** Se establece un control de flujo de datos ya sea por hardware, por software o ninguno.
- **Data Bits:** Para elegir el número de bits de datos que se transmiten en cada paquete.
- **Parity:** Si se desea utilizar bits de paridad o no.
- **Stop Bits:** Para seleccionar los bits de parada a utilizar.
- **Test/Query:** Este botón sirve para hacer un testeo del puerto COM seleccionado y la configuración de la PC.
- **Host Setup:** Permite al usuario establecer el modo con el que trabajará el módulo Xbee, esto es: en modo de comando AT (para transmisión transparente), o en modo de comandos API.

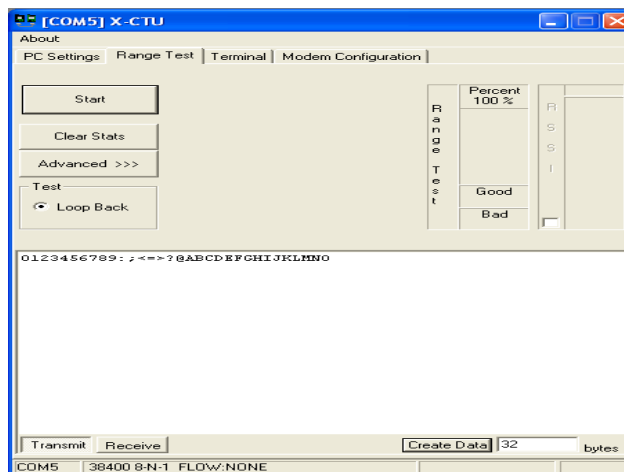


Cap.3: Fig. 22 Pestaña PC Settings y sus parámetros de configuración .

FUENTE: AUTOR

▪ RANGE TEST

En esta pestaña se comprueba el rango de radio enlace mediante el envío de un paquete de datos especificado por el usuario y verificando la respuesta del paquete mismo con el tiempo especificado.



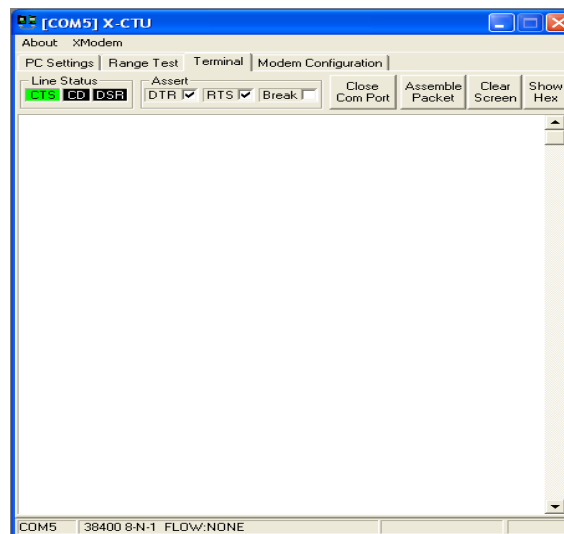
Cap.3: Fig. 23 Pestaña Range Test.

FUENTE: AUTOR.

▪ TERMINAL.

Esta pestaña permite emular una terminal, la cual se utiliza para configurar los módulos Xbee mediante comandos AT, también habilita el envío y recepción de

paquetes de datos predefinidos haciendo uso de la utilidad (Assemble Packet) y por último se envía y recibe los datos en formato hexadecimal mediante (Show Hex) como se indica en Fig.24.



Cap.3: Fig. 24 Pestaña terminal.

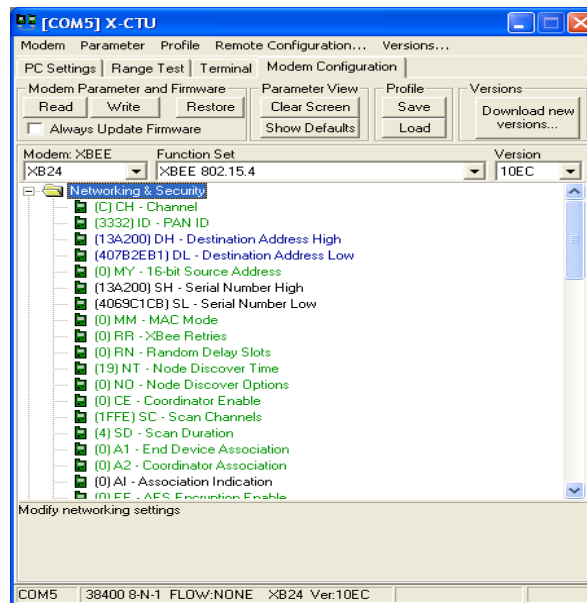
FUENTE: AUTOR.

▪ MODEM CONFIGURATION.

Esta pestaña es la última y la más importante de todas. Desempeña las siguientes funciones:

- Presenta una interfaz gráfica con el firmware del transmisor de radio frecuencia.
- Facilita la lectura o escritura de parámetros del firmware del microcontrolador del módulo.
- Se descarga de actualizaciones del firmware ya sea de la Web o bien de un archivo comprimido.
- Permite grabar o cargar un perfil de modem.

En Fig.25, se aprecia los parámetros de configuración leídos de un módulo Xbee serie1 al hacer clic en el botón "Read".



Cap.3: Fig. 25 Pestaña Modem Configuration.

FUENTE: AUTOR.

Al ejecutar la lectura del firmware de los módulos Xbee, los parámetros de configuración se presentan en tres colores distintos según sus funcionalidades:

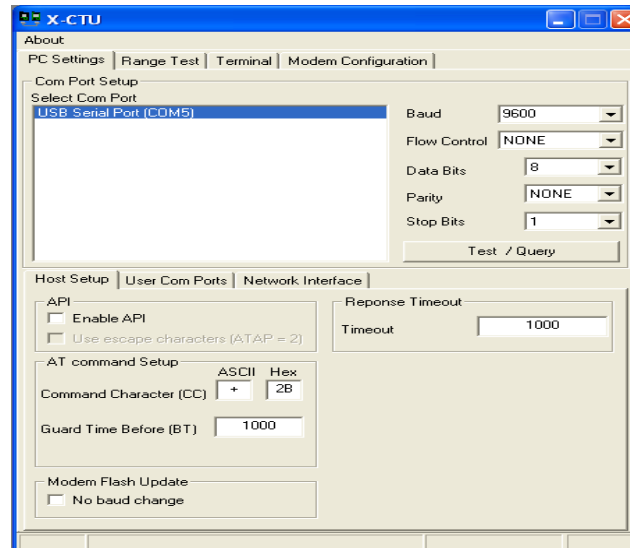
- **Negro:** Significa que el parámetro es solo de lectura y no se podrá modificar su valor nunca.
- **Verde:** Hace referencia a un parámetro cuyo valor es por defecto.
- **Azul:** Indica que el valor del parámetro ha sido modificado por el usuario.

3.2.4.2 CONFIGURACIÓN PARA UNA COMUNICACIÓN PUNTO A PUNTO.

La configuración de los módulos Xbee de la serie 1 y la serie 2 no difieren mucho entre sí, y esta diferencia radica principalmente en que a los dispositivos de la serie 2 se los puede asignar el papel que desempeñarán en la red, es decir, maestro o coordinador o esclavos o routers. Los módulos de la serie 1 no tienen esta característica o funcionalidad. En este caso se usarán Xbee de la serie 1.

Una vez conectado el módulo Xbee a un puerto USB de la computadora por medio de su respectivo adaptador, se ejecuta el aplicativo X-CTU, ver Fig.26. En el cuadro

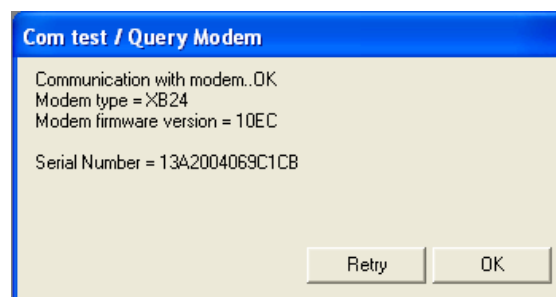
de texto de esta ventana se muestra el puerto COM emulado (COM5 en este caso) en el que se encuentra conectado el dispositivo.



Cap.3: Fig. 26 X-CTU reconoce y visualiza automáticamente los puertos disponibles en la PC.

FUENTE: AUTOR.

Para asegurar que el módulo está funcionando correctamente, se ejecuta un testeo de la conexión presionando el botón “Test/Query”; si no hay problema alguno aparecerá un cuadro de diálogo con los detalles del dispositivo, con la información del estado, la versión del firmware y el número de serie correspondiente, el cual también se encuentra impreso en el módulo Xbee. Para cerrar la ventana se hace click en “OK”.



Cap.3: Fig. 27 Detalles de testeo si el modem esta en buenas condiciones y listo para usar.

FUENTE: AUTOR.

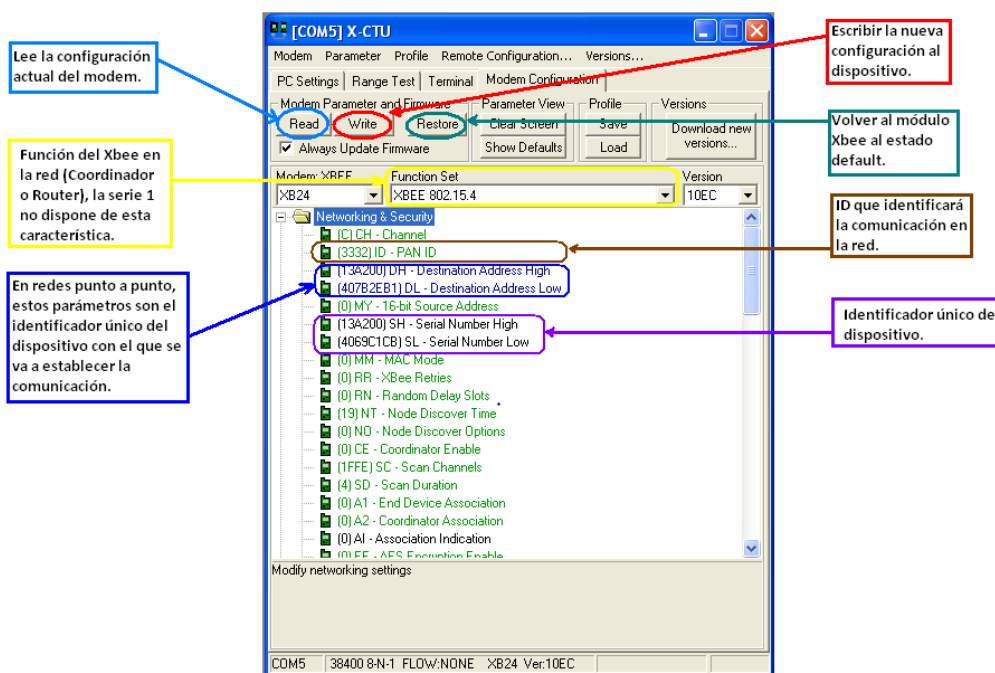
Cabe destacar que por defecto los módulos Xbee vienen configurados de fábrica a una velocidad de transmisión de 9600 baudios, si este parámetro fue cambiado y el

usuario no lo recuerda, tendrá que ir probando uno por uno hasta obtener una respuesta como se indica en Fig.27.

Existen dos métodos para configurar los dispositivos Xbee, uno es por medio de comandos AT, en una terminal como se mencionó anteriormente y el otro gráficamente. Debido a su sencillez y facilidad se usará el segundo.

Para lograr este cometido se accede a la pestaña “Modem Configuration” la cual ofrece algunas utilidades que se explican a continuación.

Con el fin de visualizar la configuración actual del modem y efectuar modificaciones, se lee desde el propio dispositivo haciendo click en el botón “Read” y se exhibe la información correspondiente, tal como se indica en Fig.28.



Cap.3: Fig. 28 Parámetros configurables para un Xbee serie 1.

FUENTE: AUTOR.

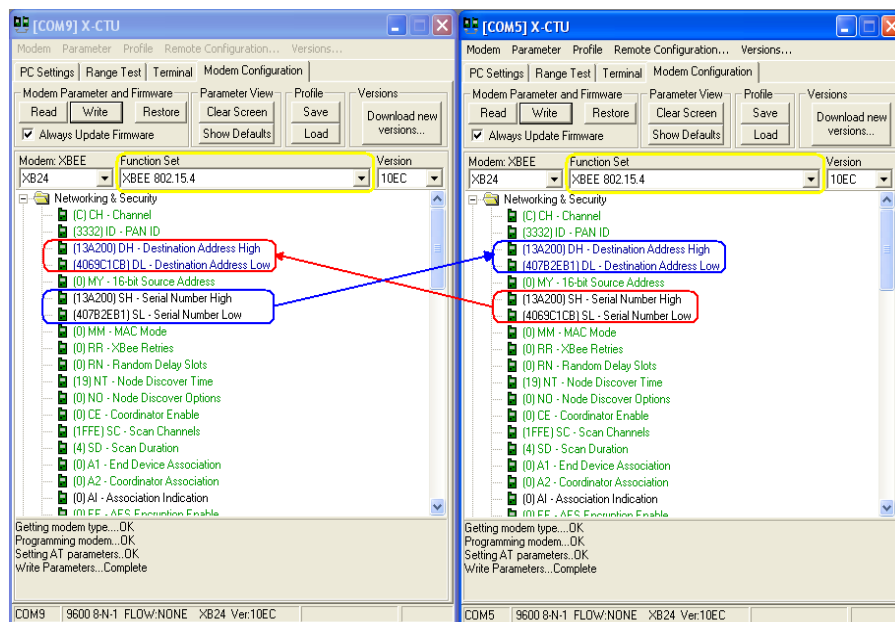
A continuación se asigna el “PAN ID” para identificar la red dentro de la cual el dispositivo Xbee estará transmitiendo o recibiendo la información, el número asignado debe estar en formato hexadecimal y dentro de un rango de 0–FFFFFFFFFFFFFFFF para la serie 2 o de 0–FFFF para la serie 1 como es el presente caso, por lo tanto cualquier módulo que quiera establecer comunicación

con la red deberá poseer el mismo “*PAN ID*”. Para este proyecto se ha elegido como identificador de red el número 3332, ver Fig.28.

El modo de configuración API o AT se elige de acuerdo a los requerimientos de la aplicación, en el modo API la información fluye en forma de paquetes, mientras que en el modo AT los datos se envían bit a bit. En este caso el tipo de configuración es indiferente ya que se trata de establecer una comunicación punto a punto. Por lo tanto se utiliza el modo de comandos AT.

En el combo de selección “*Function Set*” no se efectúa ningún cambio, dejando el parámetro preestablecido (XBEE 802.15.4) ya que como se acotó anteriormente, a los módulos de la serie 1 no se puede asignar funciones que estos desempeñarán en la red, tales como maestro o coordinador o esclavos o routers.

El siguiente paso consiste en configurar las direcciones de destino para cada uno de los dispositivos Xbee utilizando los números de serie de cada módulo. En las secciones “*Destination Address High*” y “*Destination Address Low*”, se ubican los números correspondientes del identificador del dispositivo con el que se desea establecer la comunicación, ver Fig.29.

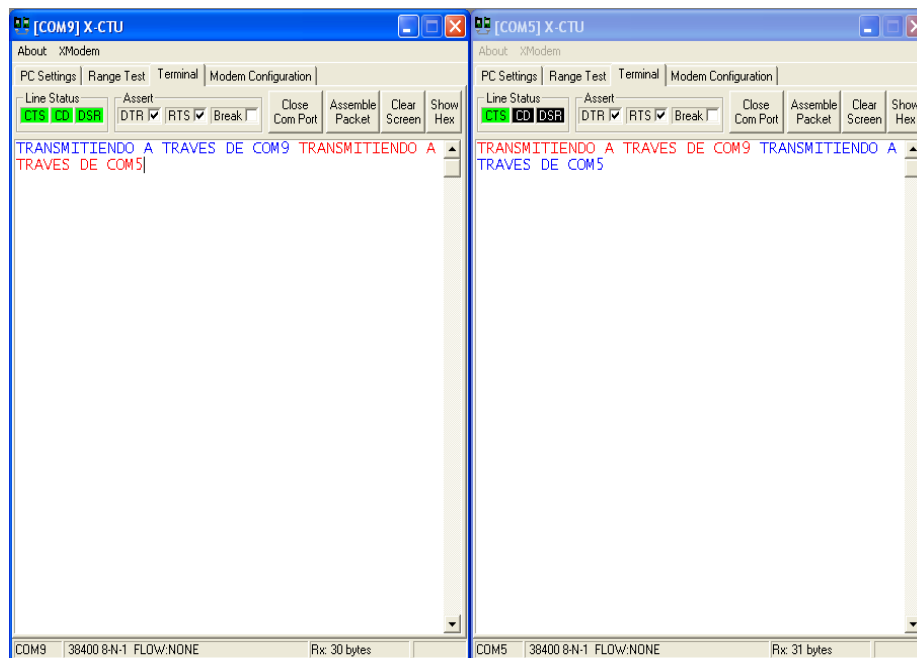


Cap.3: Fig. 29 Configuración de direcciones para dos módulos Xbee.

FUENTE: AUTOR

La nueva configuración debe ser grabada presionando el botón “Write”, si no existe ningún problema en este proceso, los mensajes emitidos por el software X-CTU deberán ser similares a los que se aprecian en la parte inferior de Fig.24.

Se comprueba que los módulos están trabajando correctamente con la nueva configuración a través de la pestaña “Terminal” enviando mensajes de un módulo a otro, ver Fig.30.



Cap.3: Fig. 30 Testeo de la comunicación entre los módulos.

FUENTE: AUTOR.

Los mensajes que se transmiten se pintan de color azul, mientras que los mensajes recibidos se pintan de color rojo.

3.2.5 QT

Qt es un entorno de desarrollo multiplataforma basado en librería Qt que permite elaborar interfaces de usuario en C++ de una manera rápida y sencilla. Fue creado por la compañía Trolltech en 1994 y en el 2008 fue adquirido por Nokia.

Inicialmente Qt fue dirigido para la creación de interfaces gráficas de usuario, actualmente existen bibliotecas para: base de datos, XML, multimedia, OpenGL.

Qt también ofrece soporte para otros lenguajes como Python mediante PyQt, Java mediante QtJambi y C# mediante Qyoto.

Los dispositivos que usan Qt son: computadoras de escritorio, teléfonos celulares, lectores electrónicos, impresoras, computadoras de automóvil, etc.

Qt está disponible bajo licencia libre de código abierto y también mediante licencia comercial cuyos cambios generados en Qt no están obligados a ser compartidos por los desarrolladores.

3.2.5.1 HERRAMIENTAS

Qt posee las siguientes herramientas que facilitan y agilizan las tareas de desarrollo:

- **Qt Assistant.** Herramienta para visualizar la documentación oficial de Qt.
- **Qt Designer.** Herramienta para la traducción de aplicaciones.
- **Qt Linguist.** Herramienta para la traducción de aplicaciones.
- **Qt Creator.** IDE para el lenguaje C++, especialmente diseñado para Qt, integra las dos primeras herramientas mencionadas.

3.2.5.2 QT CREATOR

Qt Creator es un entorno de desarrollo que incluye un editor de texto con autocompletado, diseñador de interfaces gráficas, gestión de proyectos, sistema de depuración, integración con sistemas de control de versiones.

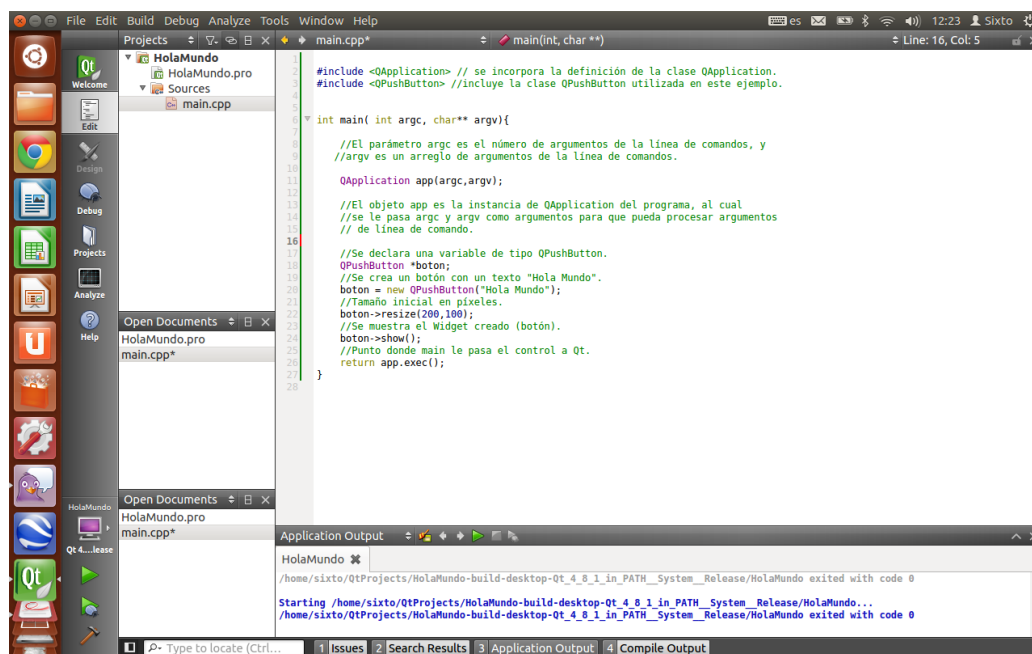
3.2.5.3 CARACTERÍSTICAS

- Editor avanzado para C++
- Herramientas potentes y sencillas para la administración y construcción de proyectos.
- Depurador visual.

3.2.5.4 INSTALACIÓN DE QT EN UBUNTU 12.04

Para instalar el framework en el sistema operativo Ubuntu 12.04, existen 4 formas distintas:

- Mediante el gestor de Synaptic se instala de una forma personalizada. Se seleccionan las librerías QT necesarias, según la demanda del desarrollador.
- Mediante la ayuda de la terminal del sistema operativo Ubuntu, para lo cual se debe ejecutar la siguiente línea de comando `sudo apt-get install qt4-dev-tools libqt4-core libqt4-gui`.
- La forma más sencilla es ejecutar al descargar el instalador desde la página de Nokia (71).



Cap.3: Fig. 31 Entorno Qt Creator.

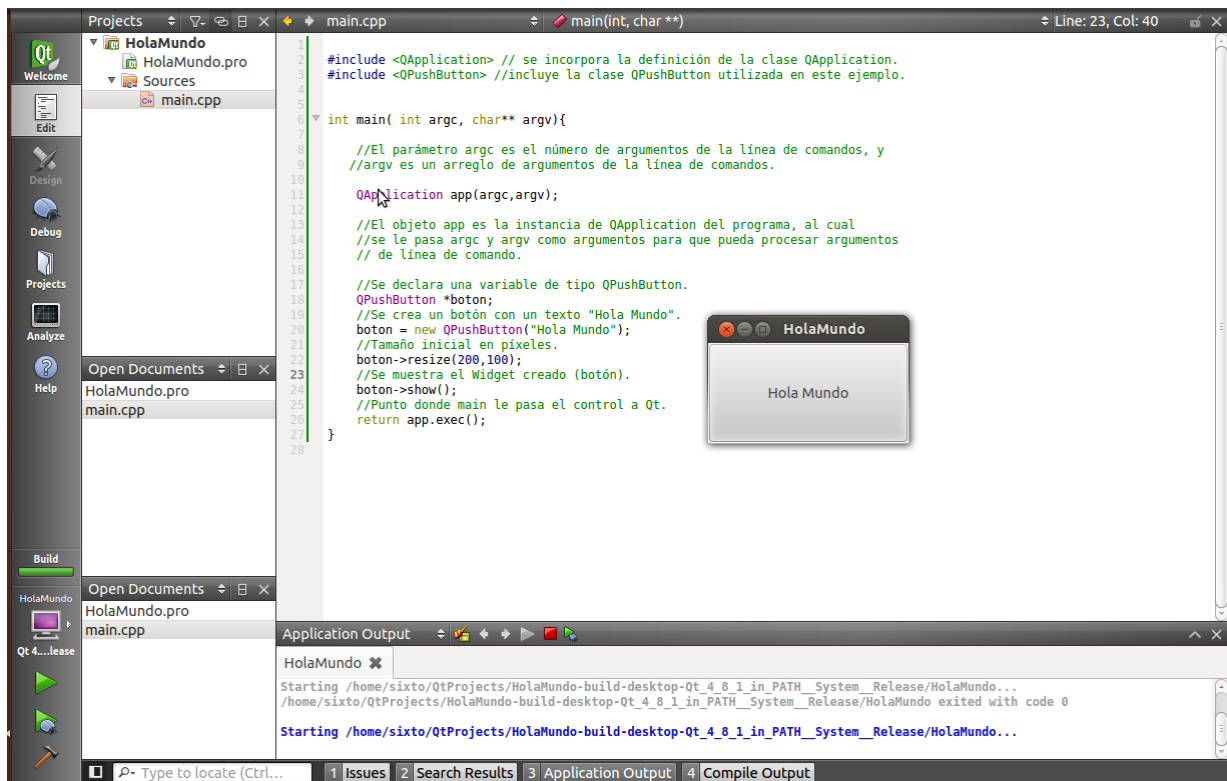
FUENTE: AUTOR.

- **Welcome.** Pantalla de bienvenida que permite cargar nuevos o últimos proyectos.
- **Edit.** Permite editar archivos de proyectos. Una barra lateral permite navegar entre archivos.

- **Debug.** Proporciona maneras para inspeccionar el estado del programa durante la depuración.
- **Projects.** Configura la forma en que los proyectos son construidos y ejecutados.
- **Output.** Examina en detalle los diversos datos, por ejemplo: construye y compila la aplicación de salida.

3.2.5.5 EJEMPLO BÁSICO

La programación de Qt Creator esta desarrollada bajo C++. Se presenta un código básico “Hola Mundo”, mas información se encuentra en la bibliografía proporcionada por el proyecto (72).



Cap.3: Fig. 32 Ejemplo básico con Qt.

FUENTE:AUTOR.

3.3 MATERIALES PARA LA CONSTRUCCIÓN DEL “GUANTE ELECTRÓNICO DE DATOS”

La estructura, el peso, la flexibilidad y en algunos casos las características técnicas electrónicas de los diferentes materiales en el desarrollo del “Guante electrónico de datos” constituyeron un valor relevante en lo técnico, ergonómico y la parte funcional del producto final.

3.3.1 GUANTE ONE POLAR.

Es el guante que reunió la mayoría de las características necesarias como base donde se montaron todos los materiales y dispositivos para el proyecto.

CARACTERÍSTICAS

- Guante para la mano derecha.
- Material de la cara dorsal, de licra con respaldo acolchado.
- Pulsera de ajuste Terry.



Cap.3: Fig. 33 Guante mano derecha Onepolar.
FUENTE: (73).

3.3.2 CONDUCTIVE THREAD - 234/34 4ply

Uno de los propósitos además de la funcionalidad óptima con la interfaz de la PC, es la presentación sencilla, evitando cables que cambien significativamente el aspecto externo del modelo original del guante; por tal motivo se eligió el denominado hilo

conductor para el transporte de las señales generadas por los sensores de contacto hacia las entradas del microcontrolador.

El material de este hilo es de plata, con una resistencia baja para ser usado en prendas de vestir.

CARACTERÍSTICAS

- Resistencia de $50\Omega/m$
- Resistividad $<0,0025 \Omega/m^2$.³³
- Diámetro nominal de 0,2 mm.



Cap.3: Fig. 34 Hilo conductor.

FUENTE: (50).

3.3.3 CONDUCTIVE FABRIC - 12"x13" MedTex130

El “Guante Electrónico de Datos” posee una serie de contactos en los dedos de la cara palmar. Para el diseño de estos contactos se eligió el material conocido como tela conductora, de esta manera se proporciona al usuario mayor comodidad al usar el guante.

CARACTERÍSTICAS

- Elástica en ambas direcciones
- Resistencia media de la superficie < 5 Ohms.
- Revestimiento 99.9% de plata pura.
- Resistencia a la abrasión de 10 000 ciclos.
- Temperatura de -30°C a 90°C .

³³ Resistividad o Resistencia específica. Constante que depende de la naturaleza del conductor.



Cap.3: Fig. 35 Tela conductora.

FUENTE: (50).

3.3.4 SNAP ASSORTMENT - (MALE AND FEMALE) AND NEEDLE

Estos corchetes permiten sujetar ó retirar elementos de una forma fácil y segura. En el desarrollo final del capítulo 4 se describe la ubicación y su uso.



Cap.3: Fig. 36 Corchetes macho y hembra.

FUENTE: (50).

3.3.5 WIRE WRAPPING WIRE

Cable aislado para unir puntos en contacto electrónico y de datos en circuitos tipo SMD, tiene un diámetro de 0,255mm.



Cap.3: Fig. 37 Cable aislado.

FUENTE: (50).

3.3.6 LITHIUM POLYMER BATTERY CELL - 3.7V 1000mAh.

La batería usada para el abastecimiento de voltaje y corriente para el sistema colocado en el guante tiene una estructura muy delgada y de peso extremadamente ligero. Cada célula emite 3,7 V a 1000mA. La batería incluye una función de protección sobre voltaje alto, sobre corriente y tensión mínima. La temperatura de funcionamiento es de entre-25C a 60C.

CARACTERÍSTICAS

- Voltaje: 3,7 V, 1000mAh de la batería.
- Capacidad: 1000mAh descarga continua.
- Química: Polímero de Litio.
- Peso: 22g.
- Conector JST (separación de 2 mm, PHR-2).
- Recargables.



Cap.3: Fig. 38 Batería polímero de litio.

FUENTE: (50).

CUIDADO DE LAS BATERÍAS POLÍMERO DE LITIO.

- No descargue los elementos por debajo de 3V durante la descarga
- Las baterías de Lipo no deben exceder 60°C durante su uso.
- Utilizar un cargador específico para esta tecnología.

3.3.7 USB LiPoly Charger - Single Cell

El cargador de baterías LIPO-USB es un circuito que está basado en el chip MCP73831T y está diseñado para recargar una sola célula LIPO 3.7V a una velocidad de 500mA o 100mA por hora.

La placa incorpora: un circuito de carga, un led indicador de estado, jumper seleccionable para la tasa de transferencia de carga que puede ser 500mA o 100mA por hora, una entrada USB, y un conector JST para batería.

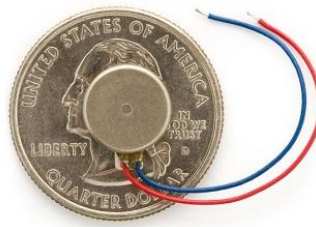


Cap.3: Fig. 39 Cargador de baterías LIPO.

FUENTE: (50).

3.3.8 VIBRATION MOTOR

Éste motor de alta calidad permite emitir una alerta no visual (vibrador) similar al funcionamiento de los teléfonos móviles. Funciona de 2 a 3,6 Voltios y ofrece una vibración no despreciable. Es usado en cualquier número de aplicaciones para indicar al usuario cuando un estado ha cambiado. Todas las partes móviles están protegidas dentro de la carcasa metálica.



Cap.3: Fig. 40 Micromotor Vibrador.

FUENTE: (50).

3.3.9 1N4007 SMD

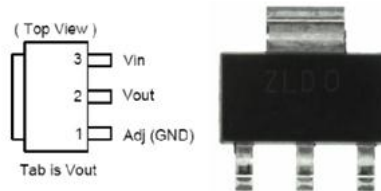
Diodo semiconductor, generalmente es usado como rectificador en fuentes de alimentación y en circuitos de protección.



Cap.3: Fig. 41 Diodo semiconductor 1N4007.
FUENTE: (74).

3.3.10 ZLDO1117 SMD

Regulador y estabilizador de voltaje a 3.3 y 5 voltios



Cap.3: Fig. 42 Regulador ZLDO1117.
FUENTE: (74).

3.3.11 TRANSISTOR NPN 2N3904 SMD



Cap.3: Fig. 43 2N3904.
FUENTE: (74).

CAPÍTULO 4

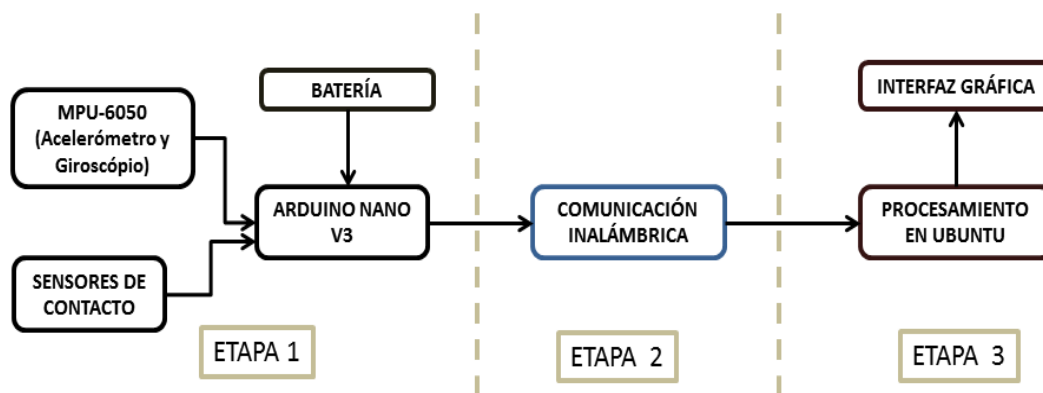
4. DESARROLLO DEL GUANTE ELECTRÓNICO DE DATOS

En este capítulo se da a conocer las partes fundamentales que comprenden el desarrollo del proyecto, en lo que involucra hardware, software y su diseño; en base a los dispositivos, materiales y conceptos fundamentales descritos en los capítulos anteriores.

Este proyecto está enfocado principalmente al uso de herramientas Open Source, utilizando la plataforma Arduino. En las páginas siguientes se detalla las características y la forma en que se utilizan sus librerías y métodos.

4.1 ARQUITECTURA DEL SISTEMA

El “Guante Electrónico de Datos” es un elemento provisto de sensores inerciales para su localización en el espacio tridimensional, también cuenta con 9 sensores de contacto cuya finalidad es la enviar comandos que serán interpretados por un programa de computadora que recibe los datos de forma inalámbrica.



Cap.4: Fig. 1 Arquitectura del sistema.

FUENTE: AUTOR.

La arquitectura del sistema se compone de tres etapas, presentadas en Fig1:

- La primera etapa constituye la fase de sensado, procesamiento y transmisión de datos hacia el primer módulo inalámbrico XBee ubicado en la parte electrónica del guante.
- La segunda etapa corresponde a la transmisión de información entre el módulo inalámbrico ubicado en el guante hacia el otro dispositivo ubicado en la computadora.
- La tercera etapa es el proceso interno en la computadora donde reside un programa que interpreta los datos recibidos.

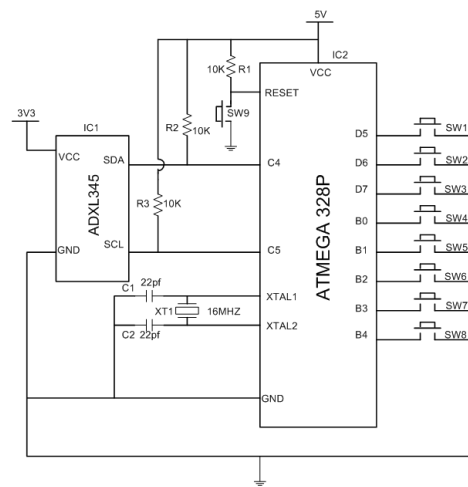
4.2 PRUEBAS INICIALES

Para la elaboración del “Guante Electrónico de Datos”, se efectuaron pruebas preliminares, con el objetivo de evaluar, verificar el funcionamiento de los dispositivos usados, detectar errores y limitaciones para realizar las correcciones respectivas.

4.2.1 PRUEBA UNO

El circuito para la prueba uno se realizó sobre un protoboard para comprobar el funcionamiento de los sensores y el sistema de comunicación, convirtiéndose en la base para el desarrollo de una segunda prueba; que llevaron a la elaboración del producto final.

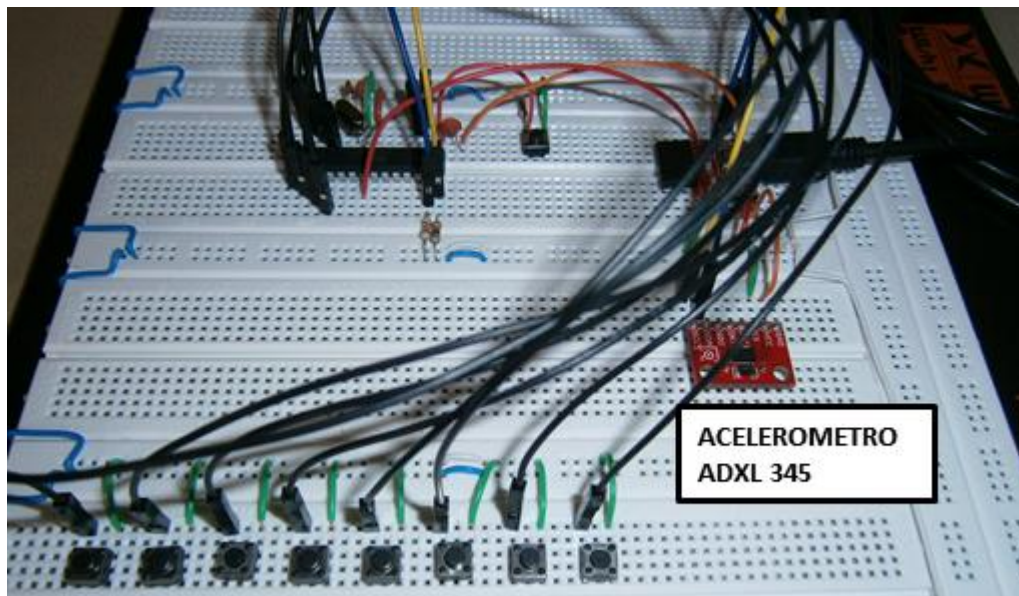
4.2.1.1 ESQUEMÁTICO PRUEBA UNO



Cap.4: Fig. 2 Esquemático Prueba Uno.
FUENTE: AUTOR.

4.2.1.2 CIRCUITO PRUEBA UNO

En Fig.3 se presenta el circuito inicial de la prueba número uno, elaborado con un microcontrolador Atmega 328p y un acelerómetro ADXL 345.



Cap.4: Fig. 3 Primer circuito en protoboard.
FUENTE: AUTOR.

4.2.1.3 ELEMENTOS USADOS

- Protoboard.
- Microcontrolador Atmega 328p.
- Acelerómetro ADXL345
- Microconvesor USB - Serial CP2102
- Pulsadores NA.
- Cable Usb a Micro Usb.

4.2.1.4 PROCEDIMIENTO

- Elaborar diagrama de conexiones según el esquemático de Fig.2.
- Montaje y conexión del acelerómetro con el microcontrolador Atmega 328p sobre un protoboard.
- Programación microcontrolador Atmega 328p.
- Comunicación con el computador.
- Observación de datos de salida.

4.2.1.5 LIMITACIONES

Luego de efectuar las pruebas de funcionamiento con la interfaz de la computadora, se determinó las siguientes limitaciones:

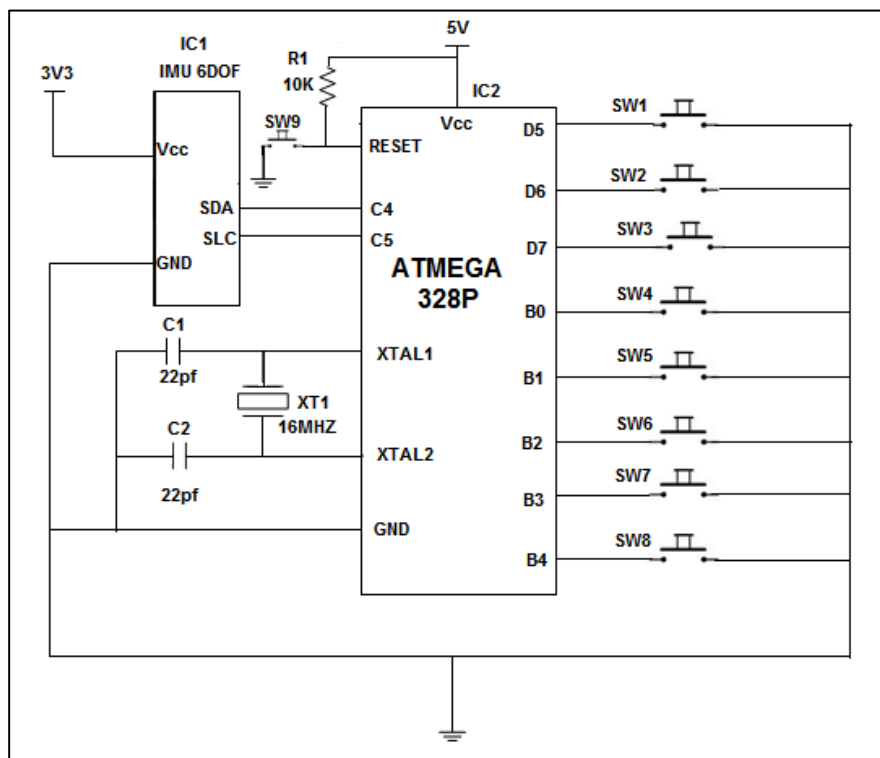
- Lecturas poco confiables por la generación de ruido, debido al uso de cables para la conexión entre el acelerómetro ADXL345 y el microcontrolador.
- El uso de componentes mecánicos (pulsadores) no permiten obtener lecturas instantáneas por efectos del rebote producido al accionarlos, tomando en cuenta que el tiempo mínimo requerido por el usuario para accionar dicho elemento es de 200 a 250 ms.

- El uso de sólo un acelerómetro limita la obtención de la posición del objeto, este dispositivo es óptimo en condiciones estáticas de movimiento, es decir, sin la presencia de una aceleración lineal en cualquiera de sus ejes.
- Discontinuidad en la transferencia de datos en la comunicación serial desde el circuito de prueba a la computadora; debido a la falta de optimización al algoritmo de transmisión y recepción.

4.2.2 PRUEBA DOS

De acuerdo a los aspectos determinados en la prueba uno se incorpora un sensor inercial de 6 grados de libertad (IMU6DOF) el cual consta de un acelerómetro y un giroscopio de 6 ejes montado sobre un tablero prefabricado.

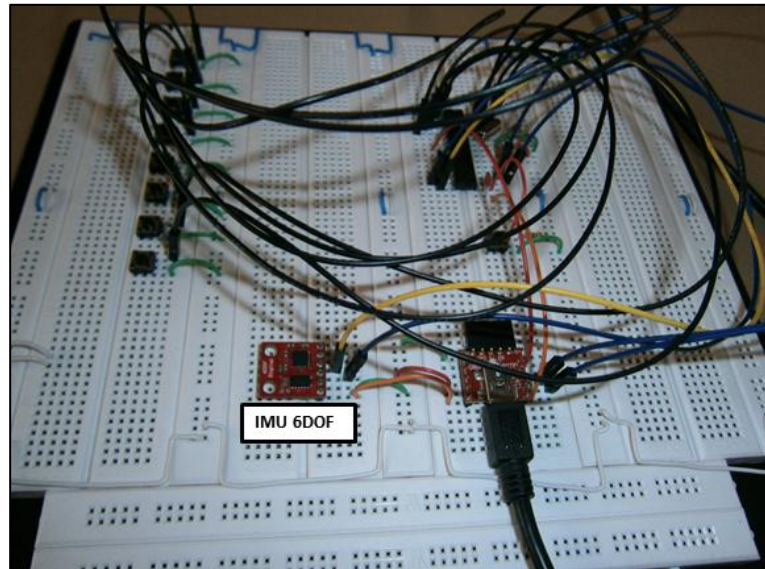
4.2.2.1 ESQUEMÁTICO DE PRUEBA



Cap.4: Fig. 4 Esquemático Prueba Dos.

FUENTE: AUTOR.

4.2.2.2 CIRCUITO DE PRUEBA



Cap.4: Fig. 5 Circuito usando IMU 6 grados de libertad.

FUENTE: AUTOR.

4.2.2.3 ELEMENTOS USADOS

- Protoboard.
- Microcontrolador Atmega 328p.
- IMU 6 grados de libertad ADXL345/ITG3200.
- Microconversor USB - Serial CP2102
- Pulsadores NA.
- Cable Usb a Micro Usb.

4.2.2.4 PROCEDIMIENTO

- Elaborar diagrama de conexiones según esquemático de Fig.4.
- Montaje y conexión de IMU 6 grados de libertad, microcontrolador Atmega 328p y pulsadores sobre protoboard.

- Programación microcontrolador Atmega 328p.
- Comunicación con el computador.
- Observación de datos de salida.

4.2.2.5 SOLUCIONES

- Los errores generados por cableado en el protoboard como se describió anteriormente se solucionan mediante la implementación de un PCB. Esta solución se hace efectiva en el producto final.
- Se implementó un algoritmo anti rebotes; para generar un tiempo de espera de 200ms en la transición de estados lógicos que se produce al accionar un pulsador. Este algoritmo se seguirá utilizando para simular un sensor de contacto hasta la elaboración del producto final, en el “Guante Electrónico de Datos” se implementarán sensores de contacto basados en textiles conductores, los cuales eliminan la generación de rebotes.
- En el capítulo dos, ítem 2.6 “Registro de la Orientación”, se determinó que mediante el uso tanto del acelerómetro y giroscopio de forma independiente, se presentan varias restricciones que hacen que el resultado final produzca errores en el sensado de su magnitud.
- La obtención de la posición basada en inclinaciones efectuadas en la prueba número dos sugiere; el uso de los dos sensores (acelerómetro y giroscopio) y fusionar las lecturas en un algoritmo de gradientes descendientes por medio de cuaterniones (capítulo dos ítem 2.6 “Registro de la Orientación”); los valores que devuelven los cuaterniones son fundamentales para traducirlos en rotaciones, permitiendo así obtener la posición del móvil de una forma más natural.
- El uso de un lazo infinito en la lectura de datos proveniente desde el circuito de pruebas hacia el computador solucionó en su mayoría el problema de discontinuidad de la lectura de datos. Este problema también se genera por otros factores que pueden ser causados por una lógica ineficiente en el algoritmo de recepción y transmisión, interrupciones de hardware y desconexión repentina del

circuito. Estos inconvenientes se solucionan mediante la implementación del PCB. El PCB se elaborará en el producto final.

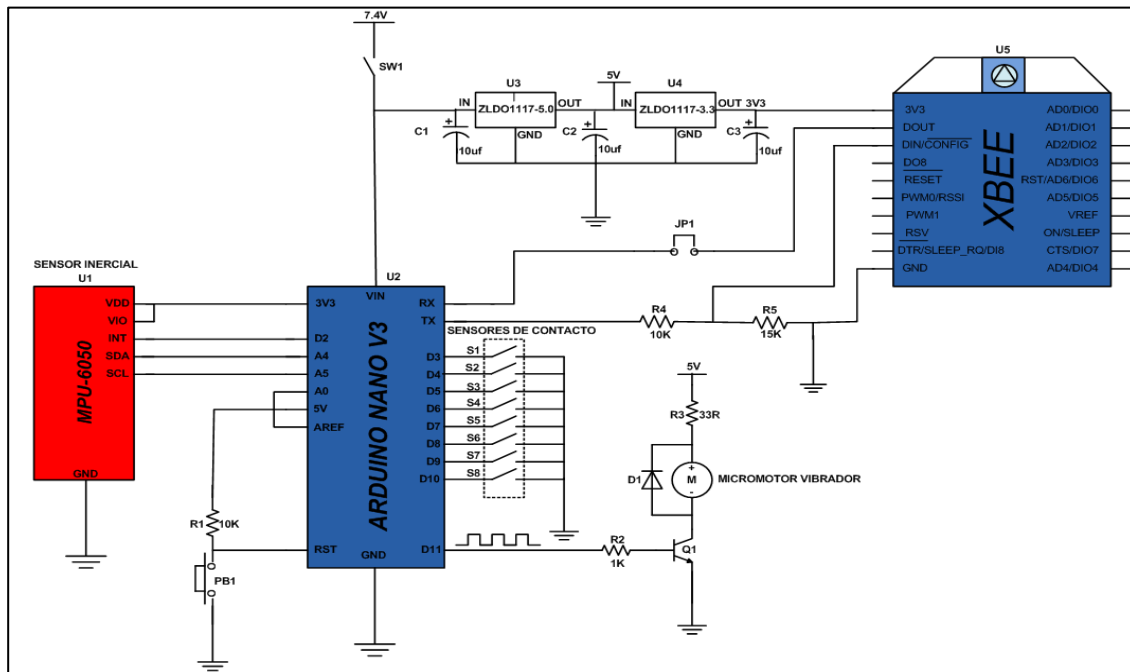
4.2.2.6 LIMITACIONES

- La IMU 6 grados de libertad está compuesta por un acelerómetro y giroscopio (separados físicamente), de tal manera que el microcontrolador se tarda una cierta cantidad de tiempo para acceder a los registros de los sensores de forma individual durante la obtención de los datos por cada ciclo de lectura. Esto provoca un desfase en la entrega de datos, considerando que este proceso debe ser coordinado en tiempo real con una aplicación que se ejecuta en el computador para llevar a cabo el movimiento del puntero del mouse y los eventos generados por los pulsadores.
- Al tiempo que se tarda el microcontrolador en tomar las lecturas se suma también; el tiempo que tarda en efectuar los cálculos matemáticos del algoritmo de gradientes descendientes para la obtención de cuaterniones.
- El uso de un cable para la comunicación serial entre el circuito de pruebas y la computadora produce cierta incomodidad que limita el libre movimiento del circuito experimental para la toma de lecturas.

4.3 PRODUCTO FINAL

Luego de realizar los diseños y pruebas preliminares mediante circuitos de ensayo, se comparó las limitaciones con los resultados obtenidos en cada circuito y mediante cambios efectuados y mejoras en cada fase anterior; se definen los elementos adecuados para el circuito final del “Guante Electrónico de Datos”.

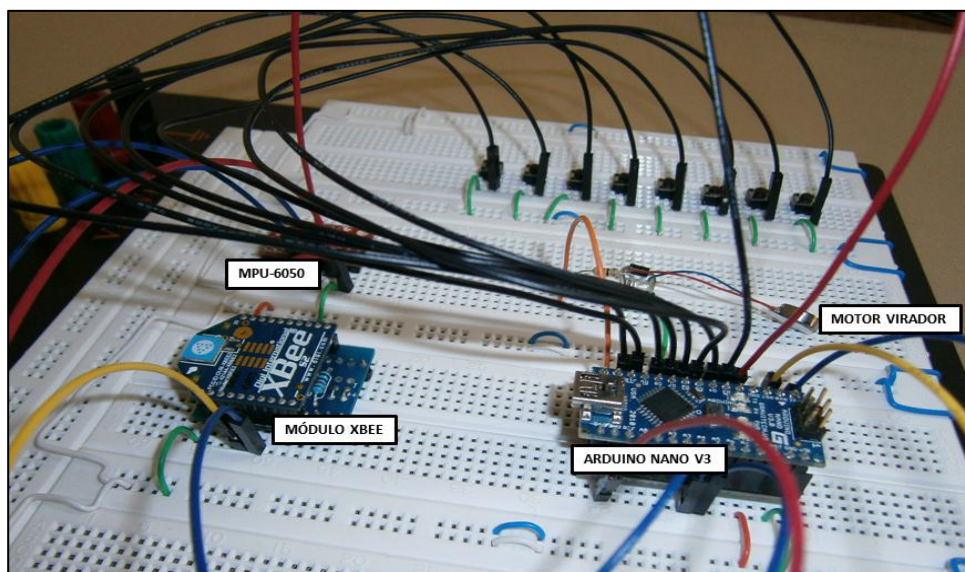
4.3.1 ESQUEMÁTICO



Cap.4: Fig. 6 Esquemático circuito final.

FUENTE: AUTOR.

4.3.2 CIRCUITO FINAL



Cap.4: Fig. 7 Circuito final “Guante Electrónico de Datos”.

FUENTE: AUTOR.

4.3.2.1 ELEMENTOS USADOS

- Protoboard.
- Sensor inercial MPU-6050.
- Digi 1mw Xbee Transceiver Module.
- Droids Sas Xbee-Usb Board.
- Arduino nano Usb microcontroller v3.
- Pulsadores NA.
- Cable Usb a Micro Usb.

4.3.2.2 SOLUCIONES

- La MPU6050 fue la mejor solución (Cap. 3 ítem 3.1.6) para resolver las limitaciones anteriores por poseer: una sola dirección I2C para la toma de las lecturas de los sensores, un procesador de movimiento digital (DMP) donde se procesan los algoritmos de filtrado y cálculos matemáticos (cuaterniones). Los datos producidos por los cuaterniones son depositados en un buffer FIFO evitando de esta manera la sobrecarga computacional del microcontrolador.
- La comunicación inalámbrica por medio de módulos XBee libera de la incomodidad del uso de cables; obteniendo mayor libertad en la manipulación del circuito.

4.3.3 GUANTE ELECTRÓNICO DE DATOS PRODUCTO FINAL.

El hardware principal es un guante diestro flexible y útil para el acoplamiento de todos los dispositivos necesarios empleados en el proyecto.

En la elección del guante para este proyecto se consideró varios parámetros detallados a continuación:

- Ergonomía.
- Flexibilidad.
- Temperatura generada por su uso

- Facilidad de montaje de los sensores, PCB y batería.

4.3.3.1 GRADOS DE LIBERTAD

El número de movimientos para concluir en una acción fue determinado de acuerdo a los requerimientos necesarios para simular las funciones de un mouse de computadora. Por tal motivo se determinó el uso de ocho sensores contactos dispuestos en las falanges distales y medias del guante y un sensor de contacto ubicado en la falange distal del dedo pulgar.



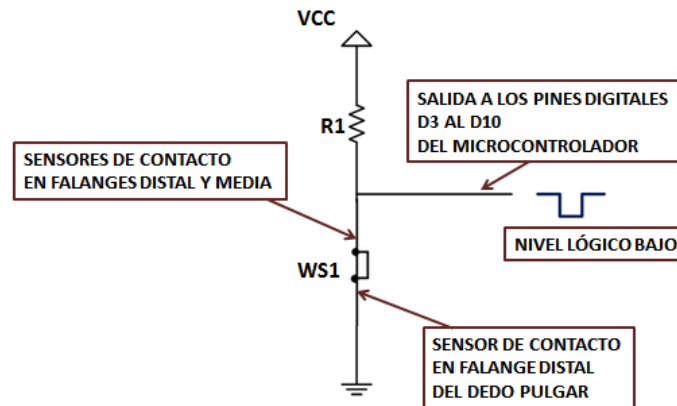
Cap.4: Fig. 8 Guante con 8 grados de libertad.

FUENTE: AUTOR

El sensado de la interacción de los cuatro dedos con el dedo pulgar, se obtiene mediante la configuración de una resistencia pull up de 10K con uno de las terminales del sensor de contacto (normalmente abierto) ubicados en cada falange distal y media de los dedos hacia las entradas digitales D3, D4, D5, D6, D7, D8, D9, D10, del módulo Arduino Nano V3.

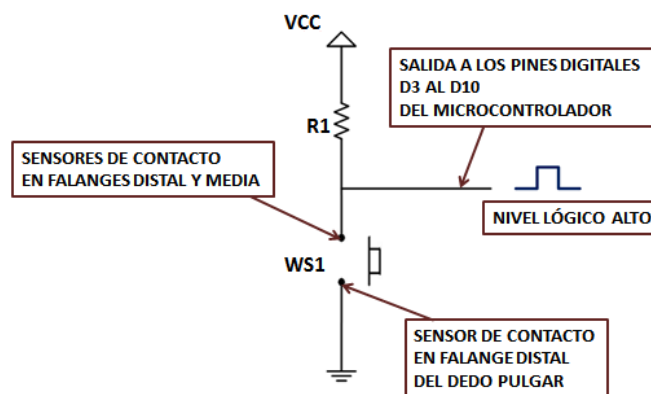
La función del sensor de contacto que se encuentra en la falange distal del dedo pulgar es la de cerrar el circuito, provocando de esta manera el flujo de corriente hacia tierra; y como consecuencia se lee en el pin correspondiente un estado lógico bajo y cuando esto no sucede se lee un estado lógico alto en dichas entradas. De tal manera que, se asocia una instrucción específica para cada contacto en forma de

comando, el mismo que es enviado hacia el computador para ejecutar una acción en concreto.



Cap.4: Fig. 9 Nivel bajo circuito cerrado por medio de los contactos.

FUENTE: AUTOR



Cap.4: Fig. 10 Nivel alto, circuito abierto.

FUENTE: AUTOR

4.3.3.2 MONTAJE DE SENSORES DE CONTACTO.

La ubicación de los dispositivos que van sobre el guante es incómoda debido al espacio reducido que se tiene, por tal motivo se eligió el uso de sensores de contacto colocados en las falanges distales y medias de los dedos de la cara palmar del guante, estas conexiones están efectuadas a través de un hilo conductor y tela conductora. Las pruebas para la ubicación de los sensores de contacto se efectuaron mediante pruebas de flexibilidad con el guante insertado en la mano.

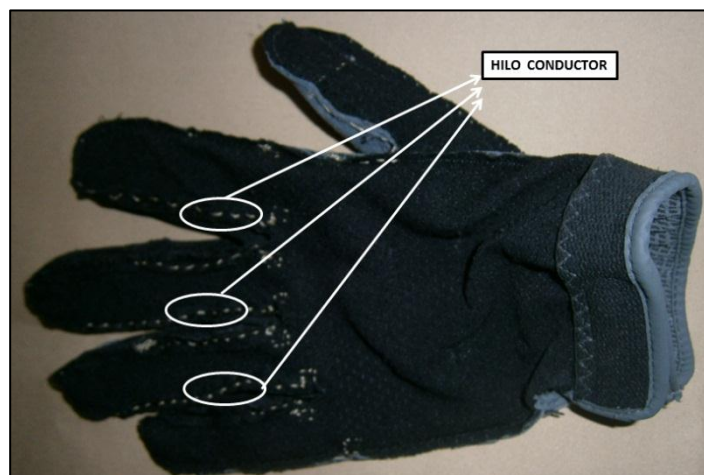


Cap.4: Fig. 11 Montaje de sensores de contacto.

FUENTE: AUTOR.

4.3.3.3 CONEXIONES INTERNAS EN EL GUANTE

Con el objetivo de concluir con un producto funcional y estético, las conexiones de los sensores de contacto ubicados en las falanges (distal y media) de los dedos del guante hacia las terminales ubicadas en las falanges proximales, están unidas mediante el hilo conductor.



Cap.4: Fig. 12 Hilo conductor en guante.

FUENTE: AUTOR.

Las terminales que llegan de los sensores de contacto están ubicadas en las falanges proximales de la cara dorsal del guante; estas son las salidas que tomarán los cables hacia las entradas digitales D3, D4, D5, D6, D7, D8, D9, D10 del Arduino Nano V3.

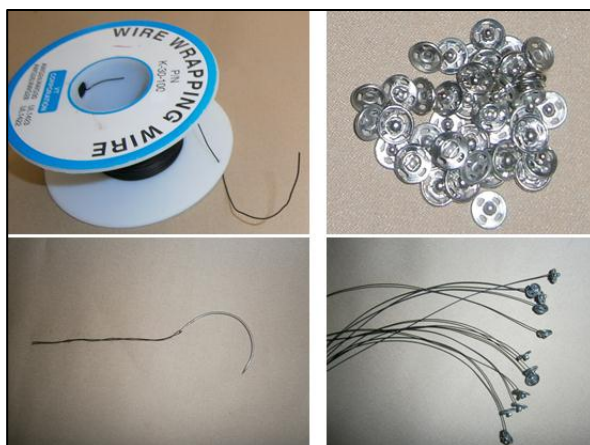


Cap.4: Fig. 13 Terminales de los sensores de contacto.

FUENTE: AUTOR.

4.3.3.4 TERMINALES DE CONEXIÓN

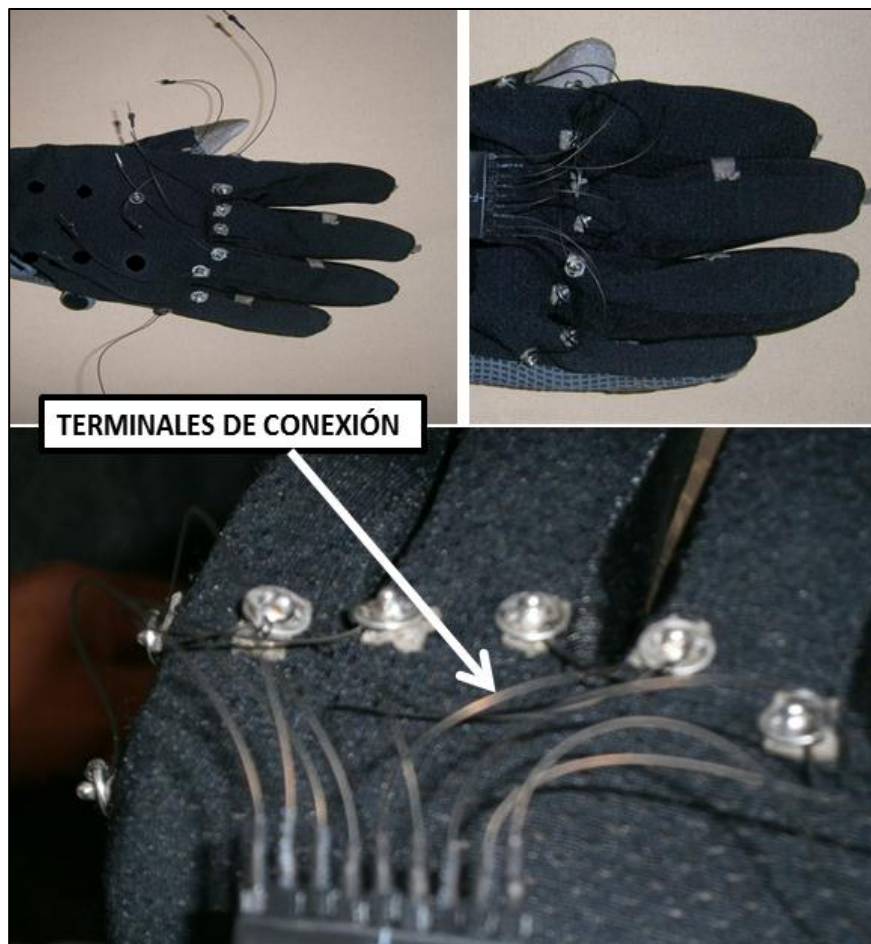
Un extremo de cada terminal se conecta a una de las salidas provenientes de los sensores de contacto. Estas terminales están armadas con el cable de conexión aislante por medio de un corchete.



Cap.4: Fig. 14 Terminales de conexión para los sensores de contacto.

FUENTE: AUTOR.

El propósito de estas terminales es la de proporcionar la comunicación entre los sensores de contacto con las entradas digitales D3, D4, D5, D6, D7, D8, D9, D10 del módulo Arduino Nano V3.



Cap.4: Fig. 15 Terminales de conexión en el guante.

FUENTE: AUTOR.

4.3.3.5 MONTAJE MOTOR VIBRADOR

El motor colocado en la parte dorsal del guante genera una vibración cuando el usuario acciona cualquiera de los sensores de contacto. Siendo esta una alternativa, ya que si no se desea el accionamiento de este motor, se la desactiva comentando una línea de código del programa principal "#define VIBRAR".

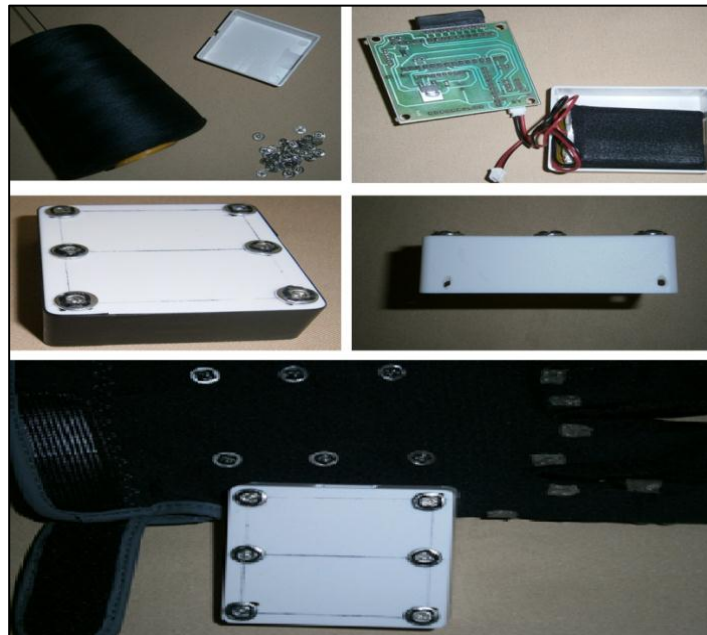


Cap.4: Fig. 16 Montaje motor vibrador.

FUENTE: AUTOR.

4.3.3.6 BASE PARA PCB Y BATERÍA.

La batería se acopla en el interior de la caja y el PCB se coloca sobre esta, así se logró incorporar: la MPU-6050, el XBee, el ArduinoNano V3 y la batería sobre la cara dorsal del guante.



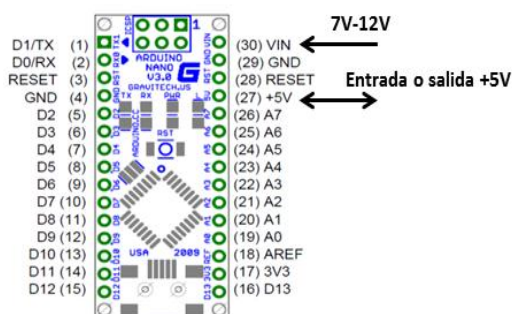
Cap.4: Fig. 17 Construcción de base para batería y PCB.

FUENTE: AUTOR.

4.3.3.7 ALIMENTACIÓN DEL CIRCUITO

Al módulo Arduino Nano V3 se alimentan mediante dos pines destinados para este propósito:

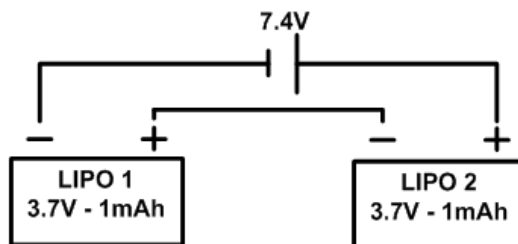
En Fig.18, el pin 27 es bidireccional es decir, ofrece +5V cuando el módulo está conectado a un puerto USB y recibe +5V regulados desde una fuente externa para alimentar al circuito cuando el tablero Arduino Nano V3 no es alimentado mediante una conexión USB.



Cap.4: Fig. 18 Distribución de Pines Arduino Nano V3.

FUENTE: (62).

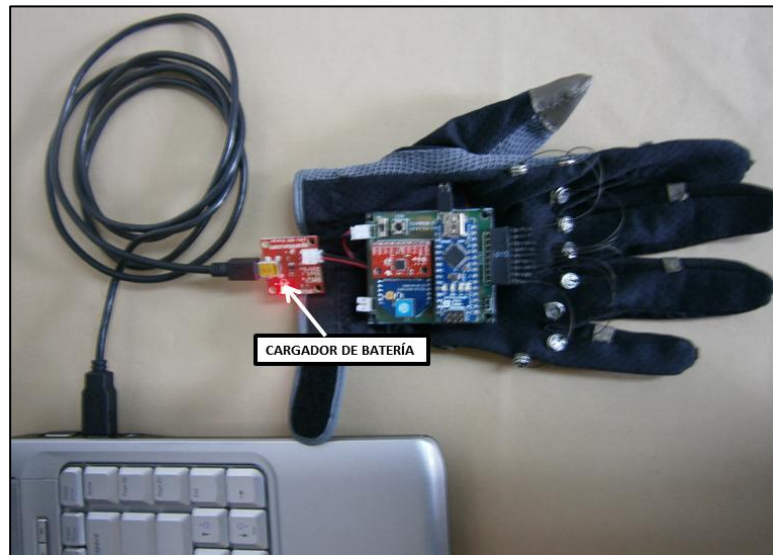
Por el pin 30 (VIN) se ingresa un rango de voltaje entre 7V a 12V para alimentar al módulo y sus periféricos; por tal motivo se eligió ejecutar la alimentación mediante esta entrada utilizando dos celdas LIPO conectadas en serie para duplicar su voltaje, obteniendo así $2 \times 3.7V = 7.4V - 1mAh$, ver Fig.19.



Cap.4: Fig. 19 Conexión serie de dos celdas LIPO.

FUENTE: AUTOR.

En estas condiciones la tarjeta Arduino Nano V3 entrega libremente 5V a través del pin 27 y 3.3V por el pin 17, voltaje que es usado para alimentar al sensor inercial MPU6050.



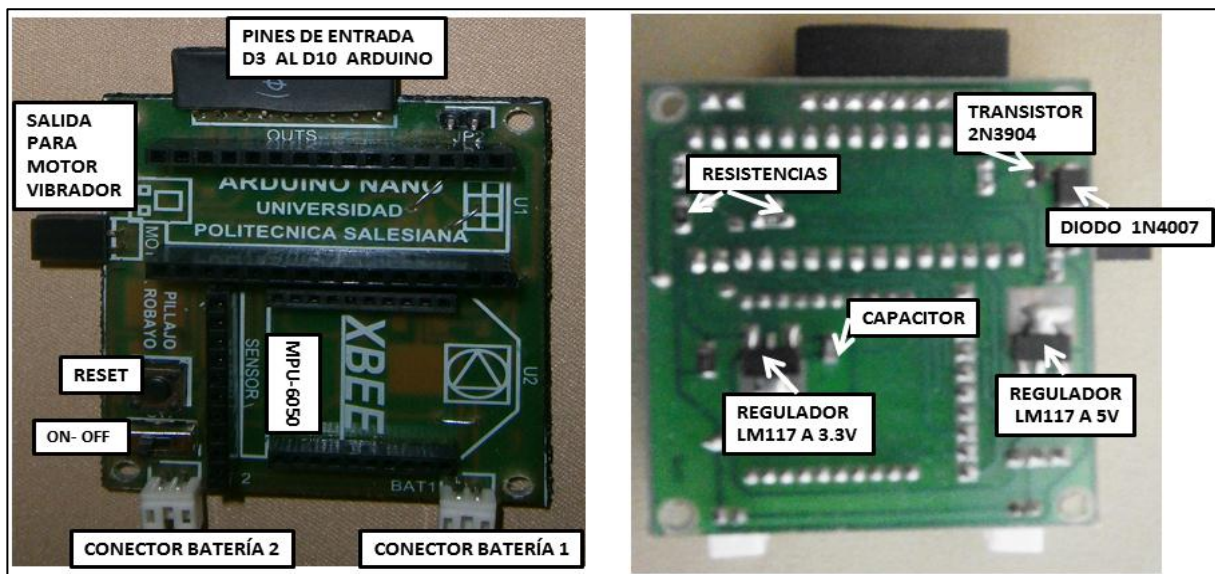
Cap.4: Fig. 20 Cargando las baterías.

FUENTE: AUTOR.

Las baterías se cargan individualmente por medio del cargador cuyas características están descritas en el capítulo 3.

4.3.3.8 PCB

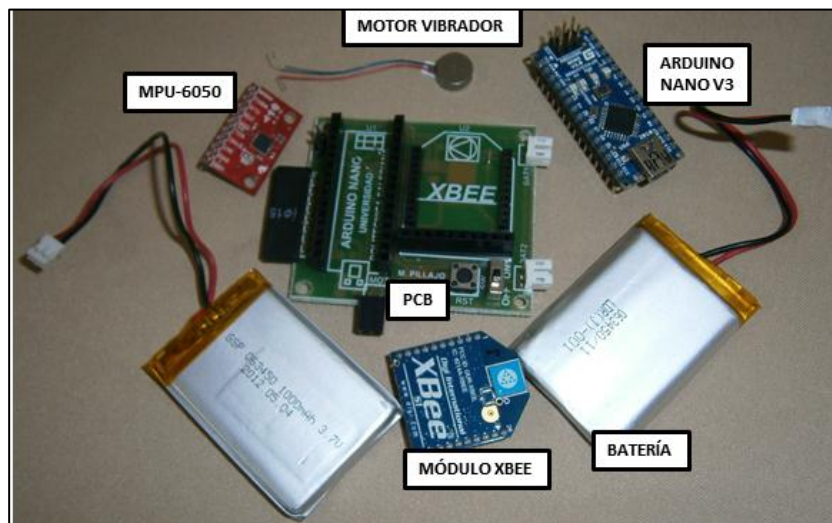
El circuito final en un PCB, basado en el diseño del esquemático de Fig.6.



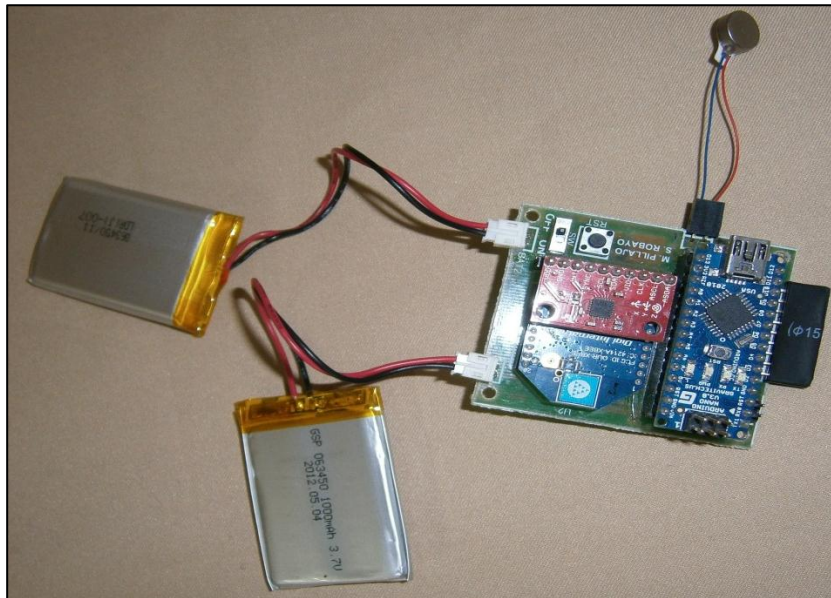
Cap.4: Fig. 21 PCB lado superior y lado inferior

FUENTE: AUTOR.

▪ MONTAJE DE ELEMENTOS SOBRE PCB

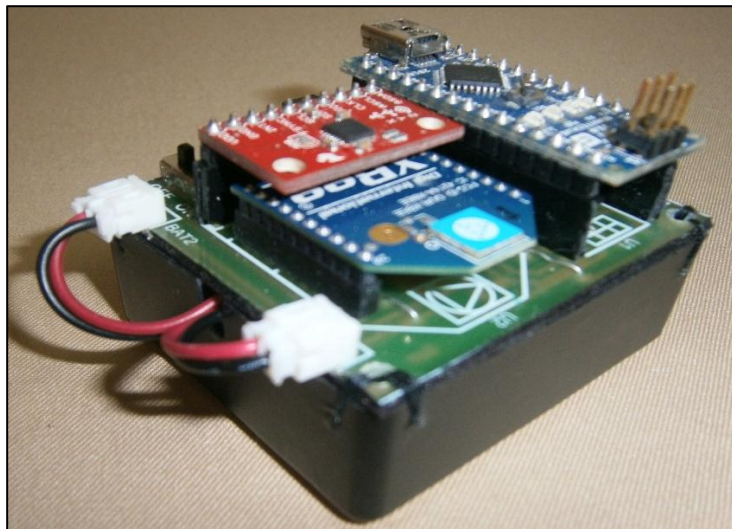
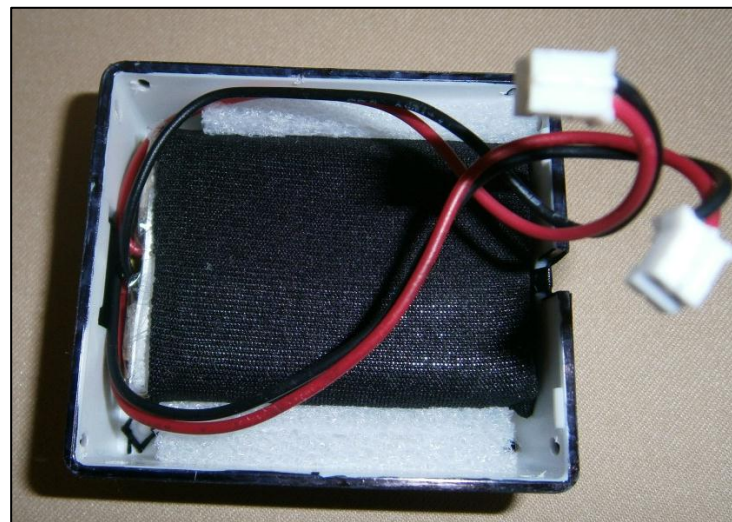


Cap.4: Fig. 22 Elementos para ser montados en PCB.



Cap.4: Fig. 23 Montaje de módulos en PCB.

FUENTE: AUTOR.



Cap.4: Fig. 24 Montaje de PCB y baterías sobre cajetín.

FUENTE: AUTOR.

4.3.3.9 LIBRERÍAS ARDUINO UTILIZADAS PARA EL GUANTE ELECTRÓNICO DE DATOS.

Arduino posee una amplia gama de librerías para ser utilizadas con diferentes propósitos. Algunas de estas bibliotecas están incluidas en el software Arduino como nativas, otras son descargadas desde una gran variedad de fuentes de terceros y ser ajustadas a un proyecto en particular.

El sensor inercial MPU-6050 es un dispositivo con altas prestaciones, por lo tanto, Arduino ofrece una biblioteca que permite el acceso a ciertos registros del dispositivo para obtener sus lecturas y puede ser descargada de su sitio Web.³⁴

Un paquete completo de librerías que ofrece muchas funcionalidades como es el acceso y operación sobre todos los registros del sensor mediante el bus I2C son las bibliotecas I2Cdev y MPU6050 desarrolladas por Jeff Rowberg³⁵, quien ha aportado significativamente a la comunidad Arduino y al desarrollo del “Guante Electrónico de Datos”.

Para la gestión de la parte inercial y la obtención de la posición del “Guante Electrónico de Datos” se utilizó las librerías MPU6050 e I2Cdev mencionadas anteriormente, la lectura y escaneo de los sensores de contacto se consiguió mediante la implementación de un código adicional.

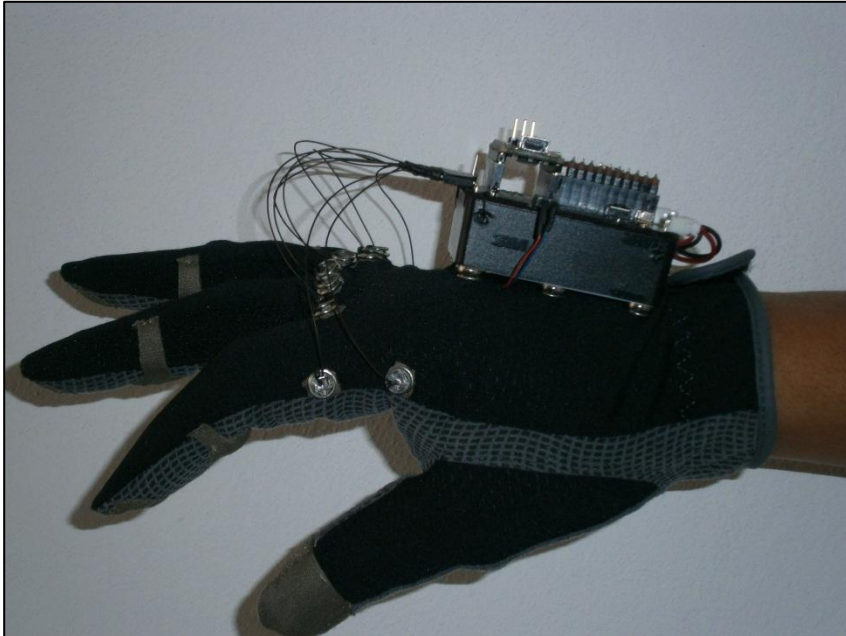
El programa Arduino donde se gestionan las referencias y llamadas a las funciones utilizadas se explica de forma detallada en el Anexo C.

³⁴ Arduino , Arduino Playground, 22-07-2012, <http://arduino.cc/playground/Main/MPU-6050>.

³⁵ Jeff Rowberg, I2C Device Library, 20-05-2012, <http://www.i2cdevlib.com/>.

4.3.3.10 GUANTE ELECTRÓNICO DE DATOS

En la cara dorsal del guante se encuentra el PCB y la batería.



Cap.4: Fig. 25 Vista frontal “Guante Electrónico de Datos”.

FUENTE: AUTOR.



Cap.4: Fig. 26 PCB en “Guante Electrónico de Datos” cara dorsal.

FUENTE: AUTOR.

Sensores de contacto ubicados en la cara palmar del “Guante Electrónico de Datos”



Cap.4: Fig. 27 Cara palmar “Guante Electrónico de Datos”.

FUENTE: AUTOR.

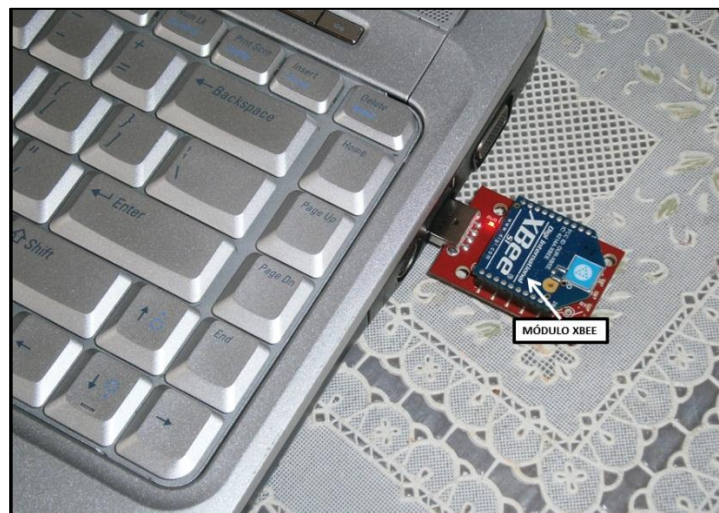
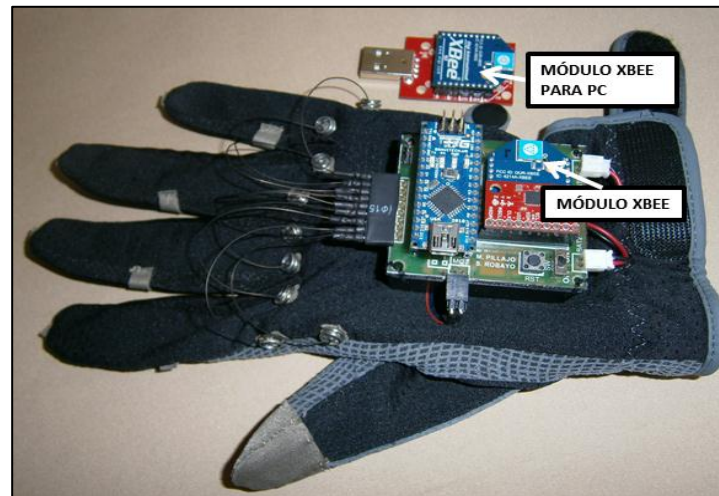


Cap.4: Fig. 28 Conexión sensores de contacto a PCB “Guante Electrónico de Datos”.

FUENTE: AUTOR.

4.3.4 PRUEBAS DE OPERACIÓN

4.3.4.1 RECEPCIÓN DE DATOS



Cap.4: Fig. 29 Módulos Xbee.

FUENTE: AUTOR.

4.3.4.2 CALIBRACIÓN Y REFERENCIA INICIAL

El proceso de calibración del sensor inercial para la obtención de la posición del “Guante Electrónico de Datos” se ejecuta de forma automática, dado que al inicializar, el sistema carga en la memoria del sensor una matriz de bytes proporcionada por la librería *mpu6050* que contiene los valores de calibración tanto para el giroscopio y el acelerómetro.

Luego de este proceso el sistema adquiere una referencia inicial la cual es muy importante ya que todo el movimiento generado por el “Guante Electrónico de Datos” es con respecto a esta posición inicial.



Cap.4: Fig. 30 Comunicación módulos Xbee.

FUENTE: AUTOR.

4.3.4.3 ENCENDIDO DEL SISTEMA

Para el uso correcto del Guante se debe seguir esta guía de pasos:

- 1) Conectar el módulo Xbee al puerto USB de la PC ver Fig.31.



Cap.4: Fig. 31 Posicionamiento del guante antes del encendido.

FUENTE: AUTOR.

- 2) Ubicar el guante en la posición presentada en Fig.32.



Cap.4: Fig. 32 Posicionamiento del guante antes del encendido.

FUENTE: AUTOR.

- 3) Encender el dispositivo y esperar el aviso mediante el diodo led de color naranja (20 segundos de espera); cuando este se ilumina el sistema esta calibrado y listo para usarse.
- 4) Correr el ejecutable del programa para el procesamiento de la información e interacción con el Guante.



Cap.4: Fig. 33 Icono de ejecutable.

FUENTE: AUTOR.

- 5) Colocarse el guante y cerrar el velcro de sujeción.
- 6) Para obtener el control del mouse se debe generar una señal de inicio mediante el contacto de las falanges distales del dedo pulgar con el dedo índice y medio ver Fig.34.



Cap.4: Fig. 34 Evento Start.

FUENTE: AUTOR.

- 7) Si el usuario desea devolver el control del mouse a la computadora deberá ejecutar un evento de parada mediante el contacto de las falanges distales del dedo pulgar con el dedo medio y anular ver Fig.35.



Cap.4: Fig. 35 Evento Stop.

FUENTE: AUTOR.

4.3.4.4 DESPLAZAMIENTOS REALIZADOS POR EL USUARIO

Para la ejecución final del movimiento del “Guante Electrónico de Datos” se requiere del movimiento total del brazo derecho; la acción final se genera desde el hombro e involucra el trabajo del codo y la muñeca.



FUENTE: AUTOR.

▪ GIRO EN YAW

Para generar un desplazamiento del cursor sobre la coordenada horizontal de la pantalla de la PC, el usuario realiza movimientos: de rotación que involucran el hombro, el codo con un movimiento de pronación, y la muñeca con movimientos de aducción y abducción.



Cap.4: Fig. 36 Movimientos para el desplazamiento horizontal del cursor.

FUENTE: AUTOR.

▪ GIRO EN PITCH

Los desplazamientos del cursor sobre la coordenada vertical del monitor de la computadora se consiguen mediante movimientos de flexión y extensión: del hombro, el codo y la muñeca.



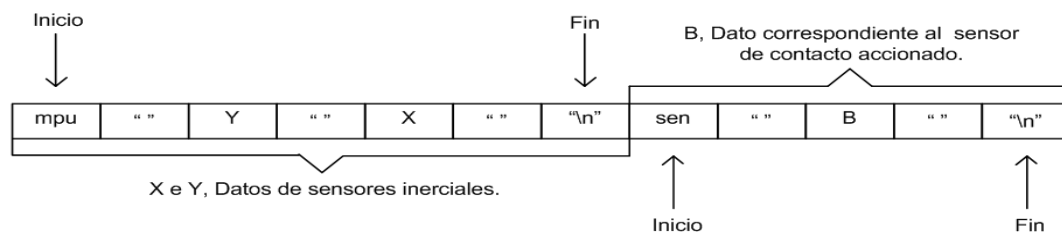
Cap.4: Fig. 37 Movimientos para el desplazamiento vertical del cursor.

FUENTE: AUTOR.

4.3.4.5 EVENTOS DE LOS SENSORES DE CONTACTO

CATEGORIZACIÓN DE DATOS

Las lecturas obtenidas por los sensores inerciales y de contacto del “Guante Electrónico de Datos” son transmitidas inalámbricamente como una cadena de caracteres. Una forma que permitió categorizar a los datos provenientes de los dos grupos de sensores fue mediante la implementación de dos comandos especiales: uno que especifica el inicio y otro el fin de la trama de datos correspondiente a cada uno de ellos. Con el fin de procesar los datos de forma individual en la PC, los caracteres contenidos en una trama están separados por espacios en blanco y se ajustan al modelo presentado en Fig.39.



Cap.4: Fig. 38 Transmisión de datos representados por cadenas de caracteres.

FUENTE: AUTOR.

Con el fin de visualizar la evolución de los datos provenientes del Guante, las salidas generadas por cada una de las acciones o eventos se consigue mediante la impresión por consola mediante el formato siguiente:

COMANDO SENSOR INERCIAL	DATO COORDENADA (X)	DATO COORDENADA (Y)	COMANDO SENSOR DE CONTACTO	DATO SENSOR DE CONTACTO
mpu	(--)	(--)	sen	(-)

Cap.4: Tabla 1 Formato de visualización de datos para cada evento.

FUENTE: AUTOR.

▪ EVENTO 1

Cuando el usuario hace contacto mediante la falange distal del dedo pulgar con la falange distal del dedo índice; el sistema envía el comando 1 en forma de caracter a través de la comunicación inalámbrica hacia la PC



COMANDO SENSOR INERCIAL	DATO COORDENADA (X)	DATO COORDENADA (Y)	COMANDO SENSOR DE CONTACTO	DATO SENSOR DE CONTACTO
mpu	800	512	sen	1

Cap.4: Tabla 2 Datos evento 1.

FUENTE: AUTOR.

▪ EVENTO 2

Cuando el usuario hace contacto mediante la falange distal del dedo pulgar con la falange media del dedo índice; el sistema envía el comando 2 en forma de caracter a través de la comunicación inalámbrica hacia la PC



COMANDO SENSOR INERCIAL	DATO COORDENADA (X)	DATO COORDENADA (Y)	COMANDO SENSOR DE CONTACTO	DATO SENSOR DE CONTACTO
mpu	760	475	sen	2

Cap.4: Tabla 3 Datos evento 2.

FUENTE: AUTOR.

▪ EVENTO 3.

Cuando el usuario hace contacto mediante la falange distal del dedo pulgar con la falange distal del dedo medio; el sistema envía el comando 3 en forma de caracter a través de la comunicación inalámbrica hacia la PC



COMANDO SENSOR INERCIAL	DATO COORDENADA (X)	DATO COORDENADA (Y)	COMANDO SENSOR DE CONTACTO	DATO SENSOR DE CONTACTO
mpu	900	345	sen	3

Cap.4: Tabla 4 Datos evento 3.

FUENTE: AUTOR.

▪ EVENTO 4

Cuando el usuario hace contacto mediante la falange distal del dedo pulgar con la falange media del dedo medio; el sistema envía el comando 4 en forma de caracter a través de la comunicación inalámbrica hacia la PC



COMANDO SENSOR INERCIAL	DATO COORDENADA (X)	DATO COORDENADA (Y)	COMANDO SENSOR DE CONTACTO	DATO SENSOR DE CONTACTO
mpu	789	400	sen	4

Cap.4: Tabla 5 Datos evento 4.

FUENTE: AUTOR.

▪ EVENTO 5

Cuando el usuario hace contacto mediante la falange distal del dedo pulgar con la falange distal del dedo anular; el sistema envía el comando 5 en forma de caracter a través de la comunicación inalámbrica hacia la PC



COMANDO SENSOR INERCIAL	DATO COORDENADA (X)	DATO COORDENADA (Y)	COMANDO SENSOR DE CONTACTO	DATO SENSOR DE CONTACTO
mpu	300	280	sen	5

Cap.4: Tabla 6 Datos evento 5.

FUENTE: AUTOR.

▪ EVENTO 6

Cuando el usuario hace contacto mediante la falange distal del dedo pulgar con la falange media del dedo anular; el sistema envía el comando 6 en forma de caracter a través de la comunicación inalámbrica hacia la PC



COMANDO SENSOR INERCIAL	DATO COORDENADA (X)	DATO COORDENADA (Y)	COMANDO SENSOR DE CONTACTO	DATO SENSOR DE CONTACTO
mpu	1020	376	sen	6

Cap.4: Tabla 7 Datos evento 6.

FUENTE: AUTOR.

▪ EVENTO 7

Cuando el usuario hace contacto mediante la falange distal del dedo pulgar con la falange distal del dedo meñique; el sistema envía el comando 7 en forma de caracter a través de la comunicación inalámbrica hacia la PC



COMANDO SENSOR INERCIAL	DATO COORDENADA (X)	DATO COORDENADA (Y)	COMANDO SENSOR DE CONTACTO	DATO SENSOR DE CONTACTO
mpu	967	187	sen	7

Cap.4: Tabla 8 Datos evento 7.

FUENTE: AUTOR.

▪ EVENTO 8

Cuando el usuario hace contacto mediante la falange distal del dedo pulgar con la falange media del dedo meñique; el sistema envía el comando 8 en forma de caracter a través de la comunicación inalámbrica hacia la PC



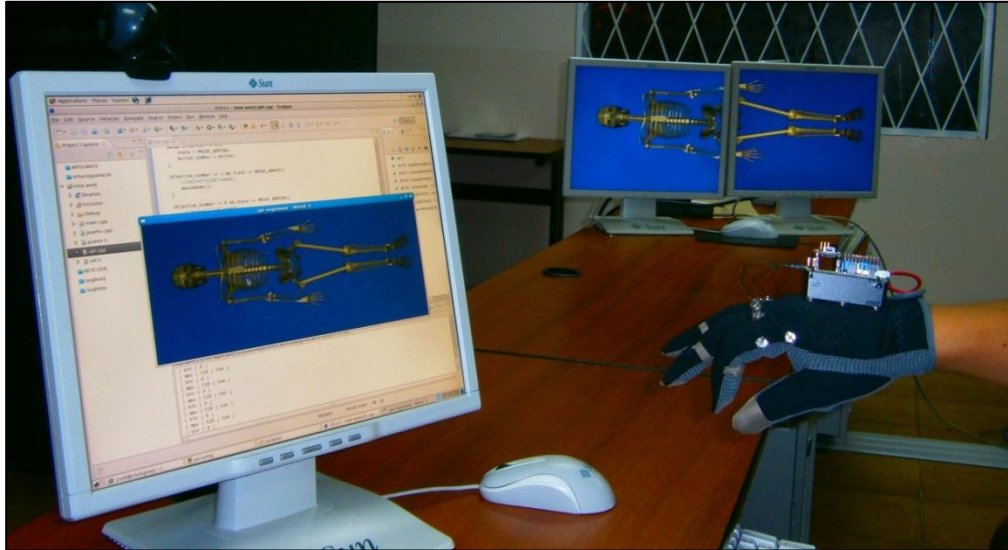
COMANDO SENSOR INERCIAL	DATO COORDENADA (X)	DATO COORDENADA (Y)	COMANDO SENSOR DE CONTACTO	DATO SENSOR DE CONTACTO
mpu	946	332	sen	8

Cap.4: Tabla 9 Datos evento 8.

FUENTE: AUTOR.

4.3.4.6 USO DEL “GUANTE ELECTRÓNICO DE DATOS”

Pruebas de operación bajo plataforma Linux.



Cap.4: Fig. 39 Prueba de operación con imagen del esqueleto humano en 3D.

FUENTE: AUTOR.



Cap.4: Fig. 40 Prueba de operación con imagen de un pie humano en 3D.

FUENTE: AUTOR.

4.3.5 COSTOS DEL PRODUCTO FINAL

ITEM	ELEMENTO	UNIDAD	COSTO
1	Arduino nano usb microcontroller v3.	1	\$32.99
2	DIGI XBee	2	\$47.90
3	XBee Explorer Dongle	1	\$24.95
5	MPU-6050	1	\$39.95
6	Guante one polar	1	\$20.00
6	Conductive thread - 234/34 4ply	1	\$34.95
7	Conductive fabric - 12"x13" MedTex130		\$29.95
8	Snap assortment - (male and female)	14	\$ 3.00
9	Wire wrapping wire	1	\$ 8.95
10	Lithium polymer battery cell - 3.7V 1000mAh.	2	\$23.90
11	USB LiPoly Charger - Single Cell	1	\$14.95
12	Vibration motor	1	\$ 4.95
13	PCB	1	\$60.00
14	Cajetín		\$ 0.90
15	2mm 10pin XBee Socket	2	\$ 2.00
16	Female Headers	2	\$ 3.00
17	Break Away Male Headers - Right Angle	1	\$ 1.95
18	JST Right-Angle Connector - Through-Hole 2-Pin	2	\$ 1.90
19	Diodo semiconductor 1N4007 SMD	1	\$ 0.20
20	Regulador ZLDO1117- 3.3 V	1	\$ 1.00
21	Regulador ZLDO1117- 5 V	1	\$ 1.00
22	Transistor NPN 2N3904 SMD	1	\$ 0.40
23	Resistencias SMD	5	\$ 1.00
24	Capacitores SMD 10 _μ F	3	\$ 1.00
25	Transporte	1	\$15.00
	TOTAL		\$380.79

5. CONCLUSIONES Y RECOMENDACIONES

5.1 CONCLUSIONES

5.1.1 CAPÍTULO UNO

- Las herramientas Open Source Hardware están poco difundidas en el medio local, siendo un conjunto de recursos basados en esquemáticos o diagramas que se encuentran a disposición de los usuarios para ser modificados, duplicados, distribuidos y finalmente transformados en un objeto tangible a conveniencia de quien requiera, abren una puerta muy importante para trabajar de manera flexible con este tipo de tecnología.
- La Realidad Virtual permite al ser humano interactuar de una forma intuitiva y natural con la computadora, proveyendo a éste una sensación de realismo y facultándole a usar sus sentidos y habilidades; de tal manera que se convierte en un complemento para el aprendizaje en muchas disciplinas.
- Los sistemas inmersos han potenciado el desarrollo de la Realidad Virtual mediante la interacción sencilla entre el usuario y la computadora ya que es posible emular al mundo real de manera casi perfecta.
- La Realidad Virtual aporta eficientemente en las prácticas de laboratorio donde ciertos procesos resultan ser peligrosos, los mismos que pueden ser simulados con esta tecnología como es el caso de impactos a altas velocidades de vehículos, contaminación, simuladores de vuelo, etc.
- Actualmente esta tecnología permite desarrollar de forma virtual; un sin número de aplicaciones antes de llevarse a la realidad; como es el caso del “Entrenamiento Virtual para Medicina”.

- Los guantes electrónicos de datos que existen actualmente en el mercado no contribuyen para el propósito del proyecto “Guante Electrónico de Datos”, sin embargo la información adquirida de la funcionalidad de estos dispositivos sirve de ayuda para plantear soluciones concretas relacionadas al desarrollo de este trabajo.
- Cuando cualquier característica funcional de la mano, ya sea mecánica o sensorial se ve afectada; las características de manipulación se reducen considerablemente, sobre todo si el daño es causado sobre el dedo pulgar y el índice, que son los ejecutores que poseen mayor grado de libertad para efectuar funciones de agarre y tacto.

5.1.2 CAPÍTULO DOS

- Un acelerómetro en estado de reposo, sobre el suelo puede medir un valor distinto de cero ya que las masas que lo conforman tienen un peso, a pesar de que no hay cambio en su velocidad. Sin embargo, un acelerómetro en caída libre medirá un valor igual a cero; porque a pesar de que su velocidad se incrementa cada vez más, esta se encuentra en un marco de referencia en el que no tiene peso.
- I2C es un protocolo sencillo e ideal para sistemas donde se maneja información entre muchos dispositivos y al mismo tiempo se requiere poco espacio y líneas de circuito impreso
- El puerto serial fue considerado por mucho tiempo como una de las conexiones básicas con un computador y un periférico, actualmente con el avance de la tecnología; la más utilizada es la conexión por puerto USB a través del uso de chips conversores USB-SERIAL, los que permiten emular un puerto serial virtual y presentar al usuario una interfaz transparente para establecer dicha conexión.
- La tecnología Zigbee es un protocolo normado que pertenece al grupo LRWPAN cuya base es el estándar IEEE 802.15.4 que rige la transmisión esporádica de

datos a baja velocidad lo cual esta óptimamente orientado a aplicaciones como el monitoreo y la automatización.

- El bajo consumo de energía en las configuraciones *ZigBee* se debe justamente a una característica fundamental que tienen estos dispositivos, que consiste en dormir durante largos períodos de tiempo, prolongando así el tiempo de vida de sus baterías.
- Zigbee está diseñado específicamente para ser la solución a problemas inalámbricos siendo una unidad pequeña capaz de adaptarse fácilmente en aplicaciones donde los cables limitan el alcance de un determinado proyecto.
- La organización ZigBee Alliance presenta a Zigbee como el nuevo estándar global para la automatización del hogar, porque permite que las aplicaciones domóticas desarrolladas por los fabricantes sean completamente interoperables entre sí, garantizando así al cliente final fiabilidad, control, seguridad y comodidad.

5.1.3 CAPÍTULO TRES

- Por asuntos comerciales y derechos de autoría la información relacionada con la tecnología con la que están elaborados los diferentes guantes de datos existentes en el mercado en lo que a software y hardware se refiere está restringida en su totalidad para el usuario. De tal manera que se decidió usar herramientas que en su mayoría brindaron la información necesaria para entender el funcionamiento de los dispositivos y sus programas; pues permitieron la utilización de sus librerías Open Source.
- El uso de Open Source Hardware permitió obtener un dispositivo acorde a las necesidades del proyecto, porque brindo la información necesaria mediante la plataforma Arduino.
- Se utilizó materiales desconocidos en el medio local, como son los textiles conductores (hilo y tela). Estos materiales aportaron significativamente en la

etapa funcional, técnica y estética; permitiendo al usuario del “Guante Electrónico de Datos” comodidad durante su manipulación.

- Para la programación del módulo Arduino Nano no se usó un programador adicional, porque el hardware del dispositivo dispone de la electrónica necesaria para este propósito, y también su microcontrolador Atmega 328P posee un gestor de arranque en su memoria denominado Bootloader, que permite cargar un nuevo código mediante una simple conexión USB.

5.1.4 CAPÍTULO CUATRO

- La incursión al tema de sensores inerciales con tecnología MEMS y la fusión, de sus datos abre un campo no conocido en detalle en el medio local, de tal manera que se presenta un gran abanico de posibilidades de generar soluciones basadas en la utilización de estos sensores.
- La plataforma Arduino utilizada como base del proyecto brindó un soporte fundamental gracias a su gran comunidad a nivel mundial tanto a nivel hardware como también software, convirtiéndose en una herramienta Open Source útil para adaptar su conocimiento e información al proyecto “Guante Electrónico de Datos”.
- Los ensayos preliminares mediante circuitos de prueba sobre un protoboard conllevaron al análisis y perfeccionamiento en cada etapa de desarrollo. Es de este modo que se descubrió limitaciones tanto a nivel hardware como software; y se implementó las mejores soluciones para la obtención del producto final.
- El módulo de comunicación se adaptó perfectamente a la plataforma Arduino, permitiendo el flujo de información inalámbrica lo más óptima posible entre el “Guante Electrónico de Datos” y la computadora.
- El funcionamiento mecánico normal de la mano constituye un movimiento en el que intervienen músculos nervios y la actuación del sistema de mando que es el cerebro. La importancia máxima en la elaboración del “Guante Electrónico de Datos” en lo que a investigación y desarrollo de todos los dispositivos se refiere;

no restó importancia al montaje de todos estos dispositivos sobre el guante, porque fue preciso ubicar todos estos elementos en un lugar donde no se disponía de mucho espacio y a su vez su uso sea lo más natural para el usuario.

- La elección del sensor inercial MPU 6050 para la obtención de la posición del “Guante Electrónico de Datos” brindó mayor estabilidad al sistema, gracias a sus sofisticadas prestaciones como son: el acondicionamiento de las señales a través de sus filtros digitales paso altos (DHPF) y paso bajos (DLPF), el procesador digital de movimiento (DMP) que gestiona toda la parte matemática reduciendo así la carga del microcontrolador y una sola dirección de lectura de datos a través de I2C.

5.2 RECOMENDACIONES

- El uso de una metodología enfocada a proyectos de desarrollo tecnológico generan un orden lógico en la ejecución de trabajos que se relacionan con la elaboración de elementos tangibles, brindando las pautas o fases para la conclusión exitosa de un trabajo.
- Este proyecto de investigación procuró poner bases en lo referente a navegación inercial, de tal manera que se pueden elaborar proyectos que se relacionen con este tipo de tecnología.
- La comunicación inalámbrica ofrece muchas ventajas a velocidades de transferencias medias o bajas, si se desea transmitir a altas velocidades, a 115200 baudios por ejemplo, los datos tienden a ser ruidosos o incoherentes, para conseguir una comunicación inalámbrica confiable se recomienda seleccionar velocidades de transferencias entre 9600 a 38400 baudios.
- Si se desea explotar todas las bondades que ofrecen los microcontroladores de Atmel AVR se recomienda programarlos usando código AVRGCC ya que estos dispositivos están fabricados para entender este lenguaje de programación, cabe

recalcar que el núcleo Arduino es un conjunto de funciones que justamente hacen llamadas a procedimientos basados en AVRGCC.

- los módulos Xbee son dispositivos bastante delicados, por lo tanto para dar un uso apropiado se recomienda ajustarse estrictamente a las especificaciones proporcionadas por el fabricante en sus respectivos data sheet. Un factor muy importante a tomar en cuenta es no exceder los límites de alimentación, ya que un voltaje superior al indicado en la hoja de datos puede provocar que el dispositivo deje de funcionar.

6. ANEXOS

6.1 ANEXO A

6.1.1 puerto.h

```
#ifndef PUERTO_H
#define PUERTO_H
#include <iostream>
#include <string>
#include <cstdlib>    //atof, atoi
#include <memory>
#include <tr1/functional>
#include <stdexcept>
#include <sstream>
#include <unistd.h>    // write(), close(), ...
#include <termios.h>   // cfsetispeed(), ...
#include <fcntl.h>     // open(), ...
#include <errno.h>
#include <sys/time.h>
#include "xutil.h"    // para simular funciones del mouse
using namespace std;
class Puerto {
public:
    Puerto(const string &puerto,uint baudios);
    virtual ~Puerto();
```

```
bool abrirPuerto(const string &puerto);  
void setVelocidad(uint velocidad);  
void setParametros();  
void getAtributos();  
void setAtributos();  
void leerPuerto();  
void cerrarPuerto();
```

```
private:
```

```
int fd;  
int estado;  
char in;  
struct termios tio;  
string bufer[50];  
string cad;  
  
XUTIL x;  
float mouseX;  
float mouseY;  
int i;  
};  
#endif // PUERTO_H
```

6.1.2 puerto.cpp

```
#include "puerto.h"  
  
Puerto::Puerto(const string &puerto, uint baudios):estado(0),mouseX(0.0f),mouseY(0.0f),i(0)  
{
```

```

//Comprobar el estado del puerto y abrir
if(puerto == "")
    throw runtime_error("Puerto no especificado");
if(!abrirPuerto(puerto.c_str())){
    throw runtime_error("No se puede abrir el puerto");
}
else{
    cout<<"Apertura del puerto OK"<<endl;
}

//Obtener los atributos del puerto
this->getAtributos();

//Configurar los parámetros de paridad, control, etc
this->setParametros();

//Configurar la velocidad en baudios
this->setVelocidad(baudios);

//Cofigurar los atributos
// Esto se debe hacer siempre después de configurar la velocidad
this->setAtributos();
}

//Cerrar el puerto
Puerto::~~Puerto() {
    this->cerrarPuerto();
}

bool Puerto::abrirPuerto(const string &puerto){
    if((this->fd = open(puerto.c_str(), O_RDWR | O_NOCTTY | O_NDELAY)) < 0){
        cerrarPuerto();
        return false;
    }
}

```

```
    }  
    else{  
        return true;  
    }  
}  
  
void Puerto::getAtributos(){  
    //Obtener los atributos del puerto  
    if(tcgetattr(fd,&tio) == estado){  
        cout<<"obtencion de atributos OK"<<endl;  
    }  
    else{  
        throw runtime_error("Error al obtener los parámetros");  
    }  
}  
  
void Puerto::setParametros(){  
    //Configuración básica para el modo no canónico  
    cfmakeraw(&tio);  
    //Habilitar la lectura e ignorar las líneas de control  
    tio.c_cflag |= CREAD | CLOCAL;  
    //configurar 8N1  
    tio.c_cflag &= ~PARENB; // sin bit de paridad  
    tio.c_cflag &= ~CSTOPB; // un bit de parada  
    tio.c_cflag &= ~CSIZE; // borrar el número de bits de datos  
    tio.c_cflag |= CS8; // ocho bits de datos  
    //Sin control de flujo por hardware  
    tio.c_cflag &= ~CRTSCTS;  
    //Sin control de flujo por software
```

```
tio.c_cflag &= ~(IXON | IXOFF | IXANY);  
tio.c_cc[VMIN] = 0;  
tio.c_cc[VTIME] = 10; //10 * 100ms = 1s  
}  
  
void Puerto::setVelocidad(uint velocidad){  
    speed_t baudios = 0;  
    switch(velocidad){  
        case 300: baudios = B300; break;  
        case 1200: baudios = B1200; break;  
        case 2400: baudios = B2400; break;  
        case 9600: baudios = B9600; break;  
        case 19200: baudios = B19200; break;  
        case 38400: baudios = B38400; break;  
        case 57600: baudios = B57600; break;  
        case 115200: baudios = B115200; break;  
    }  
    if(estado == cfsetispeed(&tio,baudios)){  
        cout<<"Velocidad de entrada ok...."<<endl;  
    }  
    else{  
        cout<<"Error en configurar la velocidad de entrada..."<<endl;  
    }  
    if(estado == cfsetospeed(&tio,baudios)){  
        cout<<"Velocidad de salida ok...."<<endl;  
    }  
    else{  
        cout<<"Error en configurar la velocidad de salida..."<<endl;  
    }  
}
```

```
        }  
    }  
void Puerto::setAtributos(){  
    //Configurar los atributos  
    // Esto se debe hacer siempre después de configurar la velocidad  
    if(estado == tcsetattr(fd,TCSANOW,&tio)){  
        cout<<"setAttrib Ok ..."<<endl;  
    }  
    else{  
        cout<<"failure setAttrib"<<endl;  
    }  
}  
void Puerto::cerrarPuerto(){  
    close(this->fd);  
}  
void Puerto::leerPuerto(){  
    int boton = 0;  
    //Limpiar el buffer de entrada  
    tcflush(fd,TCIFLUSH);  
    int result;  
    while(true){  
        result = read(fd,&in,1);  
        if(result > 0 ){  
            if((in != ' ') && (in != '\n')){  
                this->cad += char(in);  
            }  
            if(in == ' '){
```

```
    this->bufer[i] = cad;
    this->cad = "";
    i++;
}
if(in == '\n'){
    i = 0;
    string cmd = bufer[0];
    string sqy = "";
    string sqz = "";
    string strb = "";
    if(cmd == "mpu" && cad.length() <= 3){
        sqy = bufer[1];
        sqz = bufer[2];
        this->mouseY = atoi(sqy.c_str());
        this->mouseX = atoi(sqz.c_str());
        cout<<" | "<<cmd<<" | "<<mouseX<<" | "<<mouseY<<" | ";
        this->x.mouseMoveEvent(mouseX,mouseY);
    }
    if(cmd == "sen" && cad.length() <= 2){
        strb = bufer[1];
        cout<<cmd<<" | "<<strb<<" | "<<endl;
        boton = atoi(strb.c_str());
        if(boton == 9)break;
        else
            x.touchEvent(boton);
    }
}
```



```
        }//if_result
    }//while
}
```

6.2 ANEXO B

6.2.1 xutil.h

```
#ifndef XUTIL_H
#define XUTIL_H
#include <iostream>
//manejo de mouse
#include<X11/Xlib.h>
#include<X11/Xutil.h>
#include<X11/Xos.h>
#include<X11/Xatom.h>
#include<X11/keysym.h>
#include<X11/extensions/XTest.h>
#include<unistd.h> //para usar usleep(), pausa en usegundos
#define IS_DOWN 1
#define IS_UP 0
using namespace std;
class XUTIL
{
public:
    XUTIL();
    virtual ~XUTIL();
    void mouseClick();
```

```

void mouseMoveEvent(int x, int y);

void touchEvent(int button);

private:

    Display *dpy;

    Window root_window;

    int state;

};

#endif // XUTIL_H

```

6.2.2 xutil.cpp

```

#include "xutil.h"

XUTIL::XUTIL(){

    this->dpy = XOpenDisplay(NULL);

    //comprobar y abrir la conexion al servidor X
    if((this->dpy = XOpenDisplay(NULL)) == NULL){

        cout<<"Error al abrir la conexion"<<endl;

    }

    else{

        this->root_window = DefaultRootWindow(this->dpy);

    }

}

XUTIL::~XUTIL(){

    XCloseDisplay(dpy);

}

void XUTIL::mouseMoveEvent(int x, int y){

    XTestFakeMotionEvent(dpy,root_window,x,y,CurrentTime);
}

```

```
XFlush(dpy);
usleep(1000);
}
void XUTIL::touchEvent(int button){
int button_number = 0;
if(button == 0)state = IS_UP;
if(button != 0){
state = IS_DOWN;
button_number = button;
}
//clic izquierdo
if(state == IS_DOWN && button_number == 1){
XTestFakeButtonEvent(dpy,1,true,CurrentTime);
XFlush(dpy);
usleep(1000);
}
if(state == IS_UP){
XTestFakeButtonEvent(dpy,1,false,CurrentTime);
XFlush(dpy);
usleep(1000);
}
//click derecho
if(state == IS_DOWN && button_number == 2){
XTestFakeButtonEvent(dpy,3,true,CurrentTime);
XFlush(dpy);
usleep(1000);
}
```

```

if(state == IS_UP){
    XTestFakeButtonEvent(dpy,3,false,CurrentTime);
    XFlush(dpy);
    usleep(1000);
}
/*****Es posible gestionar eventos de teclado (6 opciones adicionales)*****/
/*
//Ejemplos.....
//Tecla enter
if(state == IS_DOWN && button_number == 3){
    XTestFakeKeyEvent(dpy, XKeysymToKeycode(dpy,XK_Return), true, CurrentTime);
    XFlush(this->dpy);
}
else if(state == IS_UP && button_number == 0){
    XTestFakeKeyEvent(dpy, XKeysymToKeycode(dpy,XK_Return), false, CurrentTime);
    XFlush(dpy);
}
//Tecla flecha arriba
if(state == IS_DOWN && button_number == 4){
    XTestFakeKeyEvent(dpy, XKeysymToKeycode(dpy,XK_Up), true, CurrentTime);
    XFlush(dpy);
}
else if(state == IS_UP && button_number == 0){
    XTestFakeKeyEvent(dpy, XKeysymToKeycode(dpy,XK_Up), false, CurrentTime);
    XFlush(dpy);
} */
}

```

6.3 ANEXO C

```
// =====
// ===          INCLUSIÓN DE CABECERAS          ===
// =====

//Librería nativa de Arduino necesaria para la comunicación I2C.
#include "Wire.h"

//Contiene matrices de calibración del sensor inercial y gestión del DMP (procesador digital de
movimiento).
#include "DMP.h"

//Crea un objeto para tener acceso a las funciones de la librería MPU6050.
MPU6050 mpu;

// LED, constante asociada al pin 13 del módulo Arduino Nano V3 para indicar que la calibración
//se ha llevado a cabo.
#define LED 13

// MOTOR, pin 11 para enviar pulsos PWM hacia la base el transistor que acciona el micromotor
vibrador.
#define MOTOR 11

//Comentar/descomentar para activar/desactivar la vibración al momento de accionar los sensores
//de contacto
//#define VIBRAR .

// =====

// DECLARACIÓN DE VARIABLES PARA CONTROL Y ESTADO DEL SENSOR INERCIAL MPU-
//6050

// =====

//Se pone a true si la función dmpInitialize() ha tenido éxito.
bool dmpListo = false;

//Guarda el byte que corresponde al estado de interrupción del sensor inercial MPU-6050.
```

```

uint8_t estadoInterrupcion;

//Almacena el estado luego de cada operación del dispositivo(0 = éxito, !0 = error).

uint8_t estadoMpu;

//Guarda el tamaño del paquete (por defecto 42 bytes).

uint16_t tamPaquete;

//Cuenta todos los bytes actuales en el buffer FIFO.

uint16_t fifoCount;

//Buffer para el almacenamiento de datos provenientes de la FIFO.

uint8_t fifoBuffer[64];

//Arreglo para almacenamiento de cuaterniones.

float q[4];

// =====
//          DECLARACIÓN DE VARIABLES GLOBALES          ===
// =====

bool calibrar;

bool transmitir;

bool salir;

bool aux_cal;

int actual_x;

int actual_y;

int anterior_x;

int anterior_y;

int offset_x;

int offset_y;

byte count;

byte pulso;

```

```

// =====
// ===          RUTINA PARA LA DETECCIÓN DE INTERRUPCIÓN          ===
// =====

//Bandera de interrupción, cambia de estado cada vez que hay datos disponibles en el buffer.
volatile bool mpulInterrupt = false;

//Función asociada con attachInterrupt, es llamada cada vez que hay datos disponibles en el
//buffer.

void dmpDataReady() {

    mpulInterrupt = true;

}

// =====
// ===          CONFIGURACIÓN INICIAL          ===
// =====

void setup() {

//Inicializa la comunicación I2C.

Wire.begin();

//Inicializa la comunicación serial a 38400 baudios.

Serial.begin(38400);

//Inicializa el sensor MPU-6050.

//mpu.initialize();

//Inicializa MPU-6050 y carga los parámetros de configuración al DMP y retorna un estado del
//dispositivo.

estadoMpu = mpu.dmpInitialize();

if (estadoMpu == 0) {

    // habilitar el DMP.

    mpu.setDMPEnabled(true);

    //Rutina de interrupción externa asociada a INT0 (pin D2 en Arduino Nano V3).

    // es activada cada vez que existen datos disponibles en el buffer del sensor inercial MPU-6050.

```

```
attachInterrupt(0, dmpDataReady, RISING);
estadoInterrupcion = mpu.getIntStatus();
//La inicialización del DMP ha tenido éxito.
dmpListo = true;
//Obtener el tamaño del paquete del buffer FIFO
tamPaquete = mpu.dmpGetFIFOPacketSize();
}
else {
    //La inicialización ha fallado
    Serial.print(estadoMpu);
}
//Configurar el pin 13 como salida (diodo led)
pinMode(LED, OUTPUT);
//Configurar como entrada a los pines asociados a los sensores de contacto mediante la función
//configuraPines
configuraPines();
//Inicializa la variables globales.
initGlobales();
}
//Inicializar variables globales
void initGlobales(){
    calibrar = false;
    aux_cal = true;
    transmitir = false;
    salir = false;
    actual_x = 0;
    actual_y = 0;
```



```

anterior_x = 0;

anterior_y = 0;

offset_x = 0;

offset_y = 0;

count = 0;

pulso = 170;

}

// =====

// ===          BUCLE PRINCIPAL          ===

// =====

void loop() {

//Variables para almacenar las coordenadas del mouse cuyos valores serán enviados de forma
//inalámbrica.

float mousex = 0.0f;

float mousey = 0.0f;

//Retornar al programa principal si los datos no estan listos.

if (!dmpListo) return;

    mpulInterrupt = false;

    estadoInterrupcion = mpu.getIntStatus();

    fifoCount = mpu.getFIFOCount();

if ((estadoInterrupcion & 0x10) || fifoCount == 1024) {

    mpu.resetFIFO();

    }

    else if (estadoInterrupcion & 0x02) {

//Verifica si los datos recibidos son coherentes.

while (fifoCount < tamPaquete) fifoCount = mpu.getFIFOCount();

//Lee el paquete de datos disponibles en la FIFO.

```

```

mpu.getFIFOBytes(fifoBuffer, tamPaquete);

fifoCount -= tamPaquete;

//Copia los cuaterniones de fifoBuffer al arreglo q.
mpu.dmpGetQuaternion(q, fifoBuffer);

//Almacenar las lecturas previas

anterior_x = actual_x;

anterior_y = actual_y;

//Multiplicar en cada ciclo de lectura los valor q[2] y q[3] por un factor de 5000

// para poder obtener valores más grandes y operar con ellos.

actual_y = q[2]*5000;

actual_x = q[3]*5000;

//Verificar si el sensor inercial se ha calibrado completamente y sus datos son estables.

//Esto se consigue comparando la coincidencia de 250 lecturas, en cuyo caso se activa
//"calibrar" y se considera que el dispositivo está calibrado.

    if((actual_x == anterior_x) && (actual_y == anterior_y)){

        if(count++ == 250){

            calibrar = true;

        }

    }

}

//En el momento que el sensor es calibrado,se activa el diodo led de color naranja y se
//almacena en offset_y y offset_x los valores de referencia una sola vez.

if(calibrar && aux_cal){

    digitalWrite(LED, HIGH);

    offset_y = actual_y;

    offset_x = actual_x;

    aux_cal = false;

}

```

```
//Se restan los correspondientes valores de offset de las lecturas actuales con el fin de obtener
// datos puros y se almacenan en mousex y mousey.
```

```
mousey = -(actual_y - offset_y);
```

```
mousex = -(actual_x - offset_x);
```

```
//Llamada a la función "iniciarPararTx" para verificar el estado booleano de "transmitir".
```

```
iniciarPararTx();
```

```
//Si "transmitir" es verdadero, comienza la transmisión de datos, caso contrario se suspende.
```

```
if(transmitir){
```

```
/******
```

Enviar los datos a través de puerto serial, "mpu" es un comando que sirve para indicar que los datos pertenecen al sensor inercial, separados por un espacio en blanco " " para poder capturarlos de forma individual y un final de línea "\n" para indicar el fin de la trama de datos que pertenecen a esta categoría. Se suma 425 a la coordenada Y y 640 a la coordenada X con el fin de ubicar inicialmente el puntero del mouse en el centro de la pantalla cuya resolución es 1280x800.

```
*****/
```

```
Serial.print("mpu");    Serial.print(" ");
```

```
Serial.print(mousey + 425);  Serial.print(" ");
```

```
Serial.print(mousex + 640);  Serial.print(" ");
```

```
Serial.print("\n");
```

```
//Verificar constantemente si un sensor de contacto fue presionado
```

```
escanearContactos();
```

```
}
```

```
#ifdef VIBRAR
```

```
else if(transmitir == false && salir == true){
```

```
vibrar(0);//Detener la vibración
```

```
}
```

```
#endif
```

```
}
```

```
}//loop
```

```
// "configuraPines", función que establece el papel que desempeñarán ciertos pines
```

```
//del microcontrolador del Arduino Nano.
```

```
void configuraPines(){
```

```
//Configuración pines, en los microcontroladores AVR (0 = entrada, 1 = salida)
```

```
  DDRD &= 0b00000111; // D3 - D7  entradas
```

```
  DDRB &= 0b11111000; // B0 - B2  entradas
```

```
//Se Activan las resistencias pullup internas.
```

```
  PORTD |= 0b11111000; // D3 - D7
```

```
  PORTB |= 0b00000111; // B0 - B2
```

```
}
```

```
/******
```

Función "imprimeSensor", cuyo argumento corresponde a la lectura generada por un determinado sensor de contacto accionado, el cual es enviado a través de puerto serial, "sen" es un comando que sirve para indicar que el dato pertenece a cualquiera de los 9 sensores de contacto, separados por un espacio en blanco " " para poder capturarlo de forma individual y un final de línea "\n" para indicar el fin de la trama de datos que pertenecen a esta categoría. Siempre enviará un 0 si ningún sensor de contacto es accionado.

```
*****/
```

```
void imprimeSensor(byte sensor){
```

```
  Serial.print("sen");  Serial.print(" ");
```

```
  Serial.print(sensor, DEC); Serial.print(" ");
```

```
  Serial.print("\n");
```

```
}
```

```
/******
```

Función "vibrar", recibe como argumento un valor (pulso) que está entre 0 - 255, el cual a su vez se pasa como argumento a la función "analogWrite" la cual se encarga de generar pulsos PWM por medio del pin 11 (MOTOR) usado para excitar la base del transistor y permitir el accionamiento del micromotor vibrador.

```
*****/
```

```
void vibrar(byte pulso){
```

```
  if(pulso >= 0 || pulso <= 255)
```

```

    analogWrite(MOTOR,pulso);

    else{

    Serial.println("Valor invalido...");

    return;

    }

}

/*****

```

"iniciarPararTx" es llamada en cada ciclo del lazo loop con el fin de averiguar si hay una condición de inicio/parada en la transmisión serial.

```

*****/

void iniciarPararTx(){

    if(!((PINB & 0b00000010) && !(PIND & 0b10000000))){

        transmitir = true;

    }

    if(!((PIND & 0b10000000) && !(PIND & 0b00100000))){

        transmitir = false;

        #ifdef VIBRAR

        vibrar(0);

        #endif

    }

}

/*****

```

Con la finalidad de verificar constantemente cuál de los 8 sensores de contacto es accionado, la función "escanearContactos" es llamada en cada ciclo del bucle loop. La variable "sensor" tomará diferentes valores entre 0 - 8, 0 en el caso de que ningún sensor sea accionado y un valor diferente 0 en caso contrario, de similar forma, emite señales de vibración únicamente cuando "sensor != 0"

```

*****/

```

```
void escanearContactos(){
    bool presionado = false;
    byte sensor = 0;
    if (!(PINB & 0b00000010)){ //B1
        sensor = 1;
        presionado = true;
        imprimeSensor(sensor);
#ifdef VIBRAR
        if(!salir) vibrar(pulso);
#endif
    }

    if (!(PINB & 0b00000100)){ //B2
        sensor = 2;
        presionado = true;
        imprimeSensor(sensor);
#ifdef VIBRAR
        if(!salir)vibrar(pulso);
#endif
    }

    if (!(PIND & 0b10000000)){ //D7
        sensor = 3;
        presionado = true;
        imprimeSensor(sensor);
#ifdef VIBRAR
        if(!salir)vibrar(pulso);
#endif
    }
}
```

```
#endif
}

if (!(PINB & 0b00000001)){ //B0
    sensor = 4;
    presionado = true;
    imprimeSensor(sensor);
    #ifdef VIBRAR
    if(!salir)vibrar(pulso);
    #endif
}

if (!(PIND & 0b00100000)){ //D5
    sensor = 5;
    presionado = true;
    imprimeSensor(sensor);
    #ifdef VIBRAR
    if(!salir)vibrar(pulso);
    #endif
}

if (!(PIND & 0b01000000)){ //D6
    sensor = 6;
    presionado = true;
    imprimeSensor(sensor);
    #ifdef VIBRAR
    if(!salir)vibrar(pulso);
```

```
#endif
}

if (!(PIND & 0b00001000)){ //D3
  sensor = 7;
  presionado = true;
  imprimeSensor(sensor);
#ifdef VIBRAR
  if(!salir)vibrar(pulso);
#endif
}

if (!(PIND & 0b00010000)){ //D4
  sensor = 8;
  presionado = true;
  imprimeSensor(sensor);
#ifdef VIBRAR
  if(!salir)vibrar(pulso);
#endif
}

if((!(PINB & 0b00000001)) && !(PIND & 0b01000000)){ //PB0 && PD6
  sensor = 9;
  presionado = true;
  imprimeSensor(sensor);
#ifdef VIBRAR
  if(!salir)vibrar(pulso);
```



```
    delay(200);  
    salir = true;  
#endif  
}  
  
if(!presionado){  
    sensor = 0;  
#ifdef VIBRAR  
    vibrar(0); //detener la vibración  
#endif  
    imprimeSensor(sensor);  
}  
}
```

7. BIBLIOGRAFÍA

1. **Garcia, Ortiz Frida.** Metodología de la investigación. *Aula Facil.com*. [En línea] 28 de Julio de 2007. [Citado el: 20 de Octubre de 2011.]
<http://www.aulafacil.com/cursosenviados/Metodo-Cientifico.pdf>.
2. **Rios, Wilson.** *La propiedad Inmaterial*. [En línea] 8 de Agosto de 2009. [Citado el: 14 de Marzo de 2012.] <http://revistas.uexternado.edu.co/index.php/propin/article/view/3003/2647>. 1657-1959.
3. **Coira, Xulio.** Open Hardware. *slideshare.net*. [En línea] 17 de Septiembre de 2011. [Citado el: 10 de Noviembre de 2011.] <http://www.slideshare.net/xulioc/open-hardware-9310573>.
4. **Arduino.** Arduino sitio Web. *Arduino*. [En línea] 19 de Septiembre de 2008. [Citado el: 05 de Noviembre de 2011.] <http://www.arduino.cc/es/>.
5. **Estrada, Francisco.** Diseño Gráfico y entornos virtuales. *Revista Digital Universitaria*. [En línea] 10 de Octubre de 2008. [Citado el: 22 de Noviembre de 2011.]
<http://www.revista.unam.mx/vol.9/num10/art85/art85.pdf>. 1067-6079.
6. **Reigosa, Martin.** Realidad Virtual. *SABIA Sistemas Adaptativos*. [En línea] 14 de Diciembre de 2010. [Citado el: 4 de Noviembre de 2011.]
<http://sabia.tic.udc.es/gc/Contenidos%20adicionales/trabajos/3D/Realidad%20Virtual/web/historia.html>.
7. **Ledesma, María.** Realidad virtual. *Universidad Politécnica de Madrid*. [En línea] Departamento de ingeniería electronica, 5 de Mayo de 2004. [Citado el: 6 de Noviembre de 2011.] <http://insn.die.upm.es/docs/VR0304.pdf>.

8. **Ferzola, Max.** Realidad Virtual Inmersiva. *Neoteo*. [En línea] 11 de Septiembre de 2007. [Citado el: 23 de Marzo de 2012.] <http://www.neoteo.com/en-busca-de-una-realidad-virtual-inmersiva>.
9. **Upda, Catarina.** Realidad Virtual. [En línea] [Citado el: 6 de Noviembre de 2011.] catarina.udlap.mx/u_dl_a/tales/documentos/lco/.../capitulo2.pdf.
10. **Carvallo, G.** Internet 3D y Mundos Virtuales. *ISEAMCC*. [En línea] 13 de Septiembre de 2010. [Citado el: 4 de Noviembre de 2011.] http://www.iseamcc.net/elSEA/Vigilancia_tecnologica/informe_3.pdf.
11. **Capaceti, Jorge Rubí.** Periféricos para Interacción. *Innova Tecno*. [En línea] 30 de Abril de 2010. [Citado el: 30 de Noviembre de 2011.] <http://www.innovatecno.com/Presentacion.php>.
12. **Worldviz.** Productos y servicios. *worldviz*. [En línea] 23 de Junio de 2009. [Citado el: 23 de Abril de 2011.] <http://www.worldviz.com/purchase/pricelist.php>.
13. **Tec, Biene.** Productos. *BieneTec*. [En línea] 15 de Abril de 2010. [Citado el: 30 de Noviembre de 2015.] <http://www.bienetec.es/bienetec/jsp/web/catalogo/productos/cybergloveii/index.jsp>.
14. **Reyes, Francisco.** Guantes de datos De Código Abierto. *Technology Review*. [En línea] 5 de Agosto de 2009. [Citado el: 2 de Enero de 2012.] http://www.technologyreview.es/read_article.aspx?id=35669.
15. **Kapandji, A. I.** *Fisiología Articular*. Sexta. Madrid : Panamericana, 2006. pág. 159. 84-9835-002-6.
16. **Castro, Bruno.** Anatomía, Biomecánica y Exploración del Miembro Superior. *Scrib*. [En línea] 5 de Noviembre de 2004. [Citado el: 15 de Abril de 2012.] http://es.scribd.com/teresa_martin_59/d/59794102-An-a-to-Miami-Em-Bro-Superior.

17. **Taboadela, Claudio.** Goniometría. *Scribd*. [En línea] 11 de Noviembre de 2007. [Citado el: 15 de Abril de 2012.] <http://es.scribd.com/doc/27427822/Goniometria>. 978-987-9274-04-0.
18. **Quimper, Karel.** Bimecánica del miembro Superior. *Scribd*. [En línea] 10 de Abril de 2010. <http://es.scribd.com/doc/38678053/Biomecanica-Miembro-Superior>.
19. **Ramirez.** Anatomía de la mano. *Scribd*. [En línea] 10 de Agosto de 2010. [Citado el: 25 de Enero de 2012.] <http://es.scribd.com/doc/35695943/Extremidad-Superior-Anatomia-de-la-mano>.
20. **Taylor, Graig.** The Anatomy and Mechanics of the. *Artificial Limbs*. [En línea] 6 de Mayo de 1995. [Citado el: 20 de Enero de 2012.] http://www.oandplibrary.org/al/pdf/1955_02_022.pdf.
21. **Serrano, Ana.** Curso de Física Básica. *Universidad Politécnica de Madrid*. [En línea] 17 de Mayo de 2011. [Citado el: 16 de Abril de 2012.] <http://acer.forestales.upm.es/basicas/udfisica/asignaturas/fisica/cinematica/relativotr.htm>.
22. **Muñoz, Miguel.** Principios Básicos. *Manual de vuelo*. [En línea] 19 de Agosto de 2009. [Citado el: 13 de Enero de 2012.] <http://www.manualvuelo.com/PBV/PBV15.html>.
23. **Aires, Universidad de Buenos.** Navegación Integral. *Univesidad de Buenos Aires*. [En línea] 11 de Noviembre de 2007. [Citado el: 10 de Enero de 2012.] <http://laboratorios.fi.uba.ar/lscm/espana/apuntes/Introduccion.pdf>.
24. **Martinez, Humberto.** Análisis mediante sensores de movimiento. *Universidad de Murcia*. [En línea] 13 de Marzo de 2010. [Citado el: 10 de Enero de 2012.] <http://ants.dif.um.es/~humberto/asignaturas/08ef/temas/tema3.pdf>.
25. **Félez, Jesús.** Cinemática tridimensional. *OCW Univesidad Politécnica de Madrid*. [En línea] 23 de Mayo de 2005. [Citado el: 14 de Enero de 2012.] http://ocw.upm.es/ingenieria-mecanica/simulacion-en-ingenieria-mecanica/contenidos/teoria/T14_Cinemática_Tridimensional.pdf.

26. **Sempere, Juan.** Yaw, pitch and roll. *Wikipedia*. [En línea] 19 de Mayo de 2009. [Citado el: 14 de Enero de 2012.] <http://commons.wikimedia.org/wiki/File:Plane.svg>.
27. **Ruiz, Manuel.** Cinemática del Sólido. *Aero UPM*. [En línea] 27 de Septiembre de 2010. [Citado el: 12 de Enero de 2012.] http://www.aero.upm.es/departamentos/fisica/PagWeb/asignaturas/mecanica2/mec1/M1_Te_o_02_CinSolido_handout.pdf.
28. **Castillo, G.T.D.** El teorema fundamental del algebra sobre cuaterniones. *Miscelánea Matemática*. [En línea] 11 de Marzo de 2011. [Citado el: 2012 de Marzo de 2012.] http://www.miscelaneamatematica.org/Misc29/torres_c.pdf.
29. **Mendoza.** Matemáticas de cuaterniones. *Mendomatica*. [En línea] 13 de Octubre de 2010. [Citado el: 15 de Marzo de 2012.] http://www.mendomatica.mendoza.edu.ar/nro21/Temas%20de%20Matematica_Cuaterniones_21.pdf.
30. **Baturone, Anibal Ollero.** *Robótica Manipuladores y robots móviles*. Barcelona-España : Boixareu, 2001. 84-217-1313-0.
31. **Garcia, Douglas.** Ángulos de Euler y Cuaterniones. *LCD Universidad Simón Bolívar*. [En línea] 25 de Febrero de 2009. [Citado el: 15 de Marzo de 2012.] <http://ldc.usb.ve/~alacruz/cursos/ci5321/clases/Animation/Animacion%20II.pdf>.
32. **Santamaria, Ramon.** Quaterniones. *Rayasanweb*. [En línea] 10 de Enero de 2011. [Citado el: 23 de Diciembre de 2011.] <http://www.raysanweb.com/articles/quateasy.htm>.
33. **Martinez, José Ismael.** La investigación de los Microsistemas. *UNAM.mx*. [En línea] [Citado el: 28 de Enero de 2012.] <http://www.cife.unam.mx/Programa/D16/03Ciencias/60IsmaelMartinezLopez.pdf>.
34. **Varesano, Favio.** Using Arduino for Tangible Human. *Varesano.net*. [En línea] 8 de Abril de 2011. [Citado el: 16 de Diciembre de 2011.] http://www.varesano.net/files/MoS_thesis/thesis.pdf.

35. **Fernandez, Juana.** Sensor Medidor de Aceleración. *Biblioteca Universidad de Sevilla E-Reding*. [En línea] 26 de Junio de 2008. [Citado el: 27 de Diciembre de 2011.]
<http://www.bibing.us.es/proyectos/abreproy/11638/fichero/Capitulo+4.pdf>.
36. **Sill, Robert D.** Function of Piezoelectric Accelerometers. *PCB Piezotronics*. [En línea] 1 de Abril de 2007. [Citado el: 13 de Marzo de 2012.]
http://www.pcb.com/techsupport/tech_accel.php.
37. **Juraj, George.** Some Recent Developments in Acceleration Sensors. *Measurement Science*. [En línea] 1 de Mayo de 2001. [Citado el: 10 de Enero de 2012.]
<http://www.measurement.sk/Papers3/Stein.pdf>.
38. **Centeno, Daniel Monje.** Conceptos Electrónicos en la Medida de la Aceleración. *Universidad de Sevilla*. [En línea] 09 de Septiembre de 2010. [Citado el: 15 de Noviembre de 2011.] <http://www.tav.net/transductores/acelerometros-sensores-piezoelectricos.pdf>.
39. **LabSpace.** A problem with sensors. *LabSpace*. [En línea] 15 de Marzo de 2011. [Citado el: 25 de Abril de 2012.]
http://labspace.open.ac.uk/mod/resource/view.php?id=420014&direct=1#FIG001_032.
40. **Eduardo, Jose.** Sistemas Sensoriales. *Filedén*. [En línea] 19 de Agosto de 2010. [Citado el: 15 de Diciembre de 2011.]
<http://www.fileden.com/files/2010/8/11/2939307/Robot%20Humanoide%202/RH%20SEM%20Capitulo%203.pdf>.
41. **Steane, Andrew M.** Center for Quantum Computation, University of Oxford. *University of Oxford*. [En línea] 05 de Agosto de 2011. [Citado el: 29 de Diciembre de 2012.]
http://www.physics.ox.ac.uk/users/iontrap/ams/qec/qec_ams_1.html.
42. **Moreno, Aldo.** Estabilización de un helicóptero a escala. *vargasmoreno.com*. [En línea] 27 de Agosto de 2010. [Citado el: 18 de Diciembre de 2011.]
<http://vargasmoreno.com/aldo/Tesis/Tesis.pdf>.

43. **Stańczak, Paweł.** Efecto Coriolis. *Wikipedia*. [En línea] 31 de Marzo de 2007. [Citado el: 20 de Enero de 2012.] <http://commons.wikimedia.org/wiki/File:Coriolis.JPG?uselang=es>.
44. **Roberto De Nuccio, Gang Xu.** Introduction to MEMS gyroscopes. *SOLID STATE TECHNOLOGY*. [En línea] 15 de Noviembre de 2010. [Citado el: 22 de Diciembre de 2011.] <http://www.electroiq.com/articles/stm/2010/11/introduction-to-mems-gyroscopes.html>.
45. **Johnson, Colin.** Gyroscopes adding up to next commodity MEMS. *MEMS Journal*. [En línea] 12 de Agosto de 2010. [Citado el: 30 de Enero de 2012.] <http://www.memsjournal.com/2010/08/gyroscopes-adding-up-to-next-commodity-mems.html>.
46. **Madgwick, Sebastian O.H.** An efficient orientation filter for inertial and inertial/magnetic sensor arrays. *x-io Technologies Limited*. [En línea] 30 de Abril de 2010. [Citado el: 10 de Febrero de 2012.] http://www.x-io.co.uk/res/doc/madgwick_internal_report.pdf.
47. **Welch, Greg.** The Kalman Filter. *UNC Computer Science*. [En línea] 5 de Febrero de 2012. [Citado el: 15 de Febrero de 2012.] <http://www.cs.unc.edu/~welch/kalman/index.html#Anchor-Rudolph-6296>.
48. **Morales, Omar.** La Comunicación Serial. *CIRIA Universidad de las Américas Puebla*. [En línea] 27 de Diciembre de 2003. [Citado el: 28 de Febrero de 2012.] http://catarina.udlap.mx/u_dl_a/tales/documentos/lem/morales_h_oe/capitulo3.pdf.
49. **Johnson, Shawn.** El Estándar RS232. *cursomicros.com*. [En línea] 10 de Marzo de 2009. [Citado el: 20 de Diciembre de 2011.] <http://www.cursomicros.com/avr/index.html>.
50. **SPARKFUN.** SPARKFUN. [En línea] SPARKFUN, 17 de Enero de 2010. [Citado el: 20 de Marzo de 2012.] <http://www.sparkfun.com>.
51. **Beltran, Henry.** Introducción a las familias lógicas. *OoCities.org*. [En línea] 22 de Abril de 2009. [Citado el: 16 de Febrero de 2012.] <http://www.oocities.org/henryestradabeltran/logicas.pdf>.

52. **Coquet, Eduardo.** El bus I2C. *Comunidad Electrónicos.com*. [En línea] 25 de Febrero de 2010. [Citado el: 18 de Diciembre de 2011.]
<http://www.comunidadelectronicos.com/articulos/i2c.htm>.
53. **Lopez, Marco.** Sistemas digitales. *UAM Iztapalapa*. [En línea] 20 de Noviembre de 2010. [Citado el: 18 de Diciembre de 2011.] laryc.izt.uam.mx/e-educar/claroline196/.../download.php?url.
54. **Alliance, ZeegBee.** *ZeegBee Alliance*. [En línea] 27 de Enero de 2012. [Citado el: 31 de Marzo de 2012.] <http://www.zigbee.org/>.
55. **Barneda, Iván.** ZIGBEE APLICADO A LA TRANSMISIÓN DE DATOS. *Universidad Autónoma de Barcelona*. [En línea] 15 de Septiembre de 2008. [Citado el: 10 de Septiembre de 2011.]
<http://www.recercat.cat/bitstream/handle/2072/13081/PFC%20Ivan%20Barneda.pdf?sequence=1>.
56. **Domodesk.** Todo en Domótica. *Domodesk*. [En línea] 10 de Junio de 2009. [Citado el: 05 de Diciembre de 2011.] <http://www.domodesk.com/content.aspx?co=97&t=146&c=43>.
57. **Diaz, Alejandro.** *Universidad Católica de Perú*. [En línea] 10 de Abril de 2010. [Citado el: 02 de Noviembre de 2011.] <http://tesis.pucp.edu.pe/repositorio/handle/123456789/510>.
58. **Ortega, Carlos.** Zigbee. *monografias .com*. [En línea] 28 de Julio de 2008. [Citado el: 30 de Noviembre de 2011.] <http://www.monografias.com/trabajos-pdf/zigbee/zigbee.pdf>.
59. **Vázquez, Daniel.** Protocolos por ondas portadoras y vía radio que eviten la instalación de "nuevos cables" para la domótica. *CasaDomo.com*. [En línea] 05 de 11 de 2005. [Citado el: 30 de Noviembre de 2011.]
<http://www.casadomo.com/noticiasDetalle.aspx?id=7123&c=6>.
60. **Arduino.** Arduino. *Arduino*. [En línea] 15 de Abril de 2007. [Citado el: 06 de Diciembre de 2011.] http://arduino.cc/es_old/Processing/ArduinoProcessing.

61. **Enríquez, Rafael.** UNIVERSIDAD DE CORDOVA. *Uco aulas SoftwareLibre*. [En línea] 13 de Noviembre de 2009. [Citado el: 2 de Noviembre de 2011.] http://www.uco.es/aulassoftwarelibre/wp-content/uploads/2010/05/Arduino_user_manual_es.pdf.
62. **Gravitech.** Robot Shop. *Gravitech*. [En línea] 7 de Diciembre de 2009. [Citado el: 10 de Diciembre de 2011.] http://site.gravitech.us/Arduino/NANO30/Arduino_Nano3_0.pdf.
63. **Xbee.cl.** Módulos de transmisión Inalámbrica. *Xbee.cl*. [En línea] 21 de Abril de 2010. [Citado el: 30 de Noviembre de 2011.] <http://www.xbee.cl/index.html>.
64. **InvenSense.** Data sheet. *InvenSense*. [En línea] 16 de Mayo de 2012. [Citado el: 01 de Junio de 2012.] <http://www.invensense.com/mems/gyro/mpu6050.html>.
65. **Arduino.** Descargas Arduino. *Arduino*. [En línea] 02 de Mayo de 2012. [Citado el: 23 de Junio de 2012.] <http://arduino.cc/en/Main/Software>.
66. **Fernández, Adolfo.** Conjunto de drivers para tarjetas bajo Linux . *Universidad de Oviedo*. [En línea] 20 de Junio de 2005. [Citado el: 10 de Enero de 2012.] <http://isa.uniovi.es/~vsuarez/syc/trabajopractico/Manuales%20del%20usuario.pdf>.
67. **Hp.** Xlib routines. *HP OpenVMS Systems Documentation*. [En línea] 5 de Agosto de 1991. [Citado el: 28 de Diciembre de 2011.] <http://h71000.www7.hp.com/doc/73final/5642/5642pro.html>.
68. **Toro, Amador Durán.** Manejo de Eventos. *Lenguaje de Sistemas Informáticos. Universidad de Sevilla*. [En línea] 14 de Junio de 1994. [Citado el: 28 de Diciembre de 2011.] <http://www.lsi.us.es/cursos/xlib-4.html>.
69. **IBM.** Guia de Programación AIXwindows. *Redes Linux*. [En línea] 4 de Abril de 2001. [Citado el: 29 de Diciembre de 2011.] http://www.redes-linux.com/otros_sistemas/Aix/aixwnpgd.pdf.

70. **Digi.** X-CTU Configuration & Test Utility Software. *Digi.net*. [En línea] 20 de Agosto de 2008. [Citado el: 30 de Enero de 2012.]
http://ftp1.digi.com/support/documentation/90001003_A.pdf.
71. **Nokia.** Nokia. *Qt*. [En línea] 25 de Mayo de 2012. [Citado el: 10 de Febrero de 2012.]
<http://qt.nokia.com/downloads/sdk-windows-cpp-offline>.
72. **Kdehispano.** Tutorial Qt. *KDE-Hispano*. [En línea] 26 de Diciembre de 2009. [Citado el: 15 de Abril de 2012.] <http://www.kdehispano.es/?q=content/tutorial-de-qt>.
73. **Ektelon.** Amazon. [En línea] 28 de Diciembre de 2011. [Citado el: 20 de Enero de 2012.]
http://www.amazon.com/Ektelon-Controller-Racquetball-Glove-White/dp/B003ZV294G/ref=pd_sbs_a_1.
74. **Mouser.** Application Highlight. *Mouser.com*. [En línea] 01 de Diciembre de 2011. [Citado el: 20 de Diciembre de 2011.] <http://www.mouser.com/digixbeepro/>.
75. **Atmel.** Robotshop. *RobotShop*. [En línea] Febrero de 2009. [Citado el: 20 de Noviembre de 2011.] <http://www.robotshop.com/content/PDF/datasheet-com-09261.pdf>.
76. **Shop, Robot.** RobotShop. [En línea] 5 de Mayo de 2003. [Citado el: 29 de Noviembre de 2011.] <http://www.robotshop.com/store>.
77. **Andrade, Diego.** Robótica para la enseñanza de l alfabeto. *UpsQuito*. [En línea] 5 de Octubre de 2011. [Citado el: 10 de Enero de 2012.]
http://dspace.ups.edu.ec/bitstream/123456789/1068/2/Capitulo_I.pdf.
78. **FreeBSD.** The X Window System. *FreeBSD*. [En línea] 26 de Noviembre de 2003. [Citado el: 20 de Diciembre de 2011.] <http://www.freebsd.org/doc/handbook/x-understanding.html>.

