

Diseño y Simulación de una Mano Robot con Actuadores de Nitinol para Rehabilitación



**Luis Eduardo Camayo Gómez
Daniel Ricardo Ramos Tovar**

Director: PhD Oscar Andrés Vivas Albán

Universidad del Cauca

**Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Electrónica, Instrumentación y Control
Ingeniería en Automática Industrial
Popayán, Junio de 2012**

Contenido

Lista de Figuras	i
Lista de tablas.....	iii
1. Introducción.....	1
2. La mano humana.....	3
2.1. Características, funcionalidad y agarres	3
2.2. Accidentes cerebro-vasculares	5
2.3. Terapias y sistemas de rehabilitación de mano	6
2.4. Sistemas robóticos y mecánicos para rehabilitación de mano.....	8
3. Aleaciones con memoria de forma	12
3.1. Tipos de SMA.....	13
3.1.1. El Nitinol.....	14
3.1.2. Propiedades de la aleación NiTi	15
3.2. Los SMA como actuadores	16
4. Experimentación con Nitinol	18
4.1. Experimento 1	21
4.2. Experimento 2	22
4.3. Experimento 3	22
5. Herramientas software utilizadas	23
5.1. Blender.....	23
5.2. Autodesk Inventor	24
5.3. Ogre 3D.....	25
5.4. Qt.....	26
5.5. Estructura de la aplicación	26
6. Diseño e implementación del sistema	27
6.1. Sistema de rehabilitación propuesto	27
6.2. Diseño CAD del sistema	29
6.3. Implementación del sistema mecánico	39
6.4. Tarjeta de control y sensores	39
6.4.1. Tarjeta de control	39
6.4.2. Sensores.....	41
6.5. Diseño y construcción de la tarjeta de acondicionamiento de señales	43

6.5.1.	Acondicionamiento de señales	43
6.5.2.	Diseño de la tarjeta de acondicionamiento	45
6.5.3.	Fabricación de la tarjeta de acondicionamiento.....	46
6.5.3.1.	Disipadores.....	46
6.6.	Diseño del software.....	47
6.6.1.	Diseño de la interfaz de usuario.....	48
6.6.2.	Modelado 3D de la mano	58
6.6.3.	Animación del modelo 3D de la mano en Ogre	61
6.6.4.	Programación tarjeta Arduino UNO	65
6.6.5.	Control del sistema	67
6.6.6.	Integración hardware-Interfaz	69
7.	Pruebas y resultados.....	71
7.1.	Aspectos a mejorar	78
8.	Conclusiones	81
9.	Referencias	83

Lista de Figuras

Figura 1. Estructura ósea de la mano.....	3
Figura 2. Tendones extensores de la mano.	4
Figura 3. Agarres prensiles humanos.....	4
Figura 4. HEXXOR.	8
Figura 5. FTM rehabilitation device.	9
Figura 6. The Hand Mentor.....	9
Figura 7. ARMin III.....	10
Figura 8. Amadeo: Finger and Hand Rehabilitation.....	11
Figura 9. Representación macro de la transformación de fases	13
Figura 10. Efecto simple y doble de memoria de forma	13
Figura 11. Configuraciones de empleo de SMA como actuador	17
Figura 12. SMA de Nitinol empleados.	18
Figura 13. Maqueta de experimentación.	21
Figura 14. Modelo de una mano construida en Blender.....	24
Figura 15. Ensamblaje de una mano construido en Inventor.	25
Figura 16. Mano renderizada en Ogre.....	25
Figura 17. Interfaz de usuario diseñada en Qt.	26
Figura 18. Dirección de aplicación de la fuerza del Nitinol.	28
Figura 19. Ensamblaje del robot y la mano en Inventor.	28
Figura 20. Impresora ProJet HD 3000.....	29
Figura 21. Modelo de pruebas para el ángulo de inclinación de los actuadores.....	30
Figura 22. Resultados para la articulación metacarpofalángica del dedo índice.....	31
Figura 23. Resultados para la articulación interfalángica proximal del dedo índice.	31
Figura 24. Resultados para la articulación interfalángica distal del dedo índice.	31
Figura 25. Sujetadores de los dedos.	32
Figura 26. Sistema de transmisión de fuerza para los dedos 2 al 5.....	32
Figura 27. Soportes para la muñeca y el antebrazo.....	33
Figura 28. Cojinete Igubal EFOM - Modelo real y modelo CAD.	33
Figura 29. Base para los cojinetes.	34
Figura 30. Fijación del Nitinol al sistema.	34
Figura 31. Flexo-extensión del pulgar.	35
Figura 32. Barra para el movimiento del pulgar.....	35
Figura 33. Mano espástica.	36
Figura 34. Sistema completo implementado en Inventor.	38
Figura 35. Vistas del sistema fabricado.....	39
Figura 36. Tarjeta Arduino Seleccionada	41
Figura 37. Encoder incremental de dos canales.	41
Figura 38. Vistas 3D del sistema con una regleta perforada y encoder incremental.	42
Figura 39. Potenciómetro deslizante de 100K.....	42
Figura 40. Potenciómetros ensamblados en Inventor.	43
Figura 41. Esquemático de un divisor de tensión 0-5 VDC	43
Figura 42. Esquemáticos de acondicionamiento.	44
Figura 43. Verificación experimental de las etapas de acondicionamiento.	44
Figura 44. Diseño esquemático elaborado.....	45
Figura 45. Tarjeta construida: independiente (a) e integrada (b)	46
Figura 46. Grafica normalizada resistencia interna Vs temperatura.....	46
Figura 47. Multímetro con sensor de temperatura.....	47
Figura 48. TabWidget usado en la aplicación.....	50

Figura 49. Pestaña de inicio.	51
Figura 50. Opciones de inicio de sesión.	51
Figura 51. Ventana para crear un nuevo paciente.	52
Figura 52. Creación de un archivo de texto para el nuevo paciente.	52
Figura 53. Cuadro de diálogo para cargar pacientes.	53
Figura 54. Pestaña de diagnóstico.	54
Figura 55. Cuadro para modificar el diagnóstico de los pacientes cargados.	54
Figura 56. Visualización de los datos almacenados en el archivo de un paciente.	55
Figura 57. <i>Group Boxes</i> para crear y cargar ejercicios.	55
Figura 58. Selección del tipo de secuencia y del orden de activación de los dedos.	56
Figura 59. Guardar un ejercicio nuevo.	56
Figura 60. Selección del ejercicio a ejecutar.	56
Figura 61. Vista previa de uno de los ejercicios de la base de datos.	56
Figura 62. Selección de la duración de la terapia y del tiempo en extensión y flexión de los dedos.	57
Figura 63. Pestaña para programar los ejercicios de la terapia.	57
Figura 64. Visualización virtual de la terapia.	58
Figura 65. Malla y esqueleto modelados en Blender.	59
Figura 66. Llamado al exportador de mallas en Blender.	60
Figura 67. Ogre Meshes Exporter.	60
Figura 68. Sistema de coordenadas en Ogre y Blender.	61
Figura 69. Configuración de las preferencias de exportación.	61
Figura 70. Renderizado de la malla por defecto y la importada en Ogre.	63
Figura 71. Esquema de control ON-OFF.	68
Figura 72. Banda de Histéresis generada para una configuración establecida.	68
Figura 73. Diagrama de flujo representando el paso de mensajes entre la interfaz y la tarjeta Arduino UNO.	69
Figura 74. Dispositivo implementado.	71
Figura 75. Comparación entre el mecanismo real y la simulación en Inventor.	73
Figura 76. Montaje de la mano de la paciente sobre el sistema.	74
Figura 77. Ajuste de los soportes para el brazo de la paciente.	75
Figura 78. Creación de un nuevo paciente para la sesión.	75
Figura 79. Valores diagnosticados para el rango de movimiento de la paciente.	75
Figura 80. Extensión del dedo índice.	76
Figura 81. Abducción máxima del pulgar.	76
Figura 82. Comparación entre la abducción máxima del sistema y la abducción ideal del pulgar.	77
Figura 83. Secuencia ejecutada durante la terapia.	77
Figura 84. Representación virtual del movimiento de la mano del paciente.	78

Lista de tablas

Tabla 1. Rango de movimiento de las articulaciones de los dedos.....	7
Tabla 2. Composición química y propiedades de algunas aleaciones SMA	14
Tabla 3. Propiedades físicas, mecánicas y de forma de una aleación NiTi	15
Tabla 4. Características eléctricas Nitinol.....	19
Tabla 5. Especificaciones térmicas aleación Ni-Ti.	19
Tabla 6. Propiedades físicas y térmicas muelle de Nitinol empleado.....	20
Tabla 7. Resultados experimentales para un alambre de Nitinol de 0,38 mm.	21
Tabla 8. Resultados de experimentar con dos muelles de Nitinol.	22
Tabla 9. Rango de movimiento articular para el sistema.....	30
Tabla 10. Piezas diseñadas para el mecanismo.	36
Tabla 11. Especificaciones para controlar el sistema propuesto.....	40
Tabla 12. Especificaciones tarjeta Arduino UNO.....	40
Tabla 13. Temperaturas en el IRF530 con y sin disipador.	47
Tabla 14. Procedimientos y funciones implementados en Arduino UNO.	66
Tabla 15. Promedio de tiempos empleados por el sistema.	72
Tabla 16. Medidas de las manos de los pacientes de prueba.....	72
Tabla 17. Comparación entre el sistema fabricado y un sistema con motores eléctricos. .	81

1. Introducción

Los accidentes cerebro-vasculares (ACV) son eventos desafortunados que pueden ocurrirle a cualquier persona sin importar su edad, y los efectos que producen en sus afectados dependen de muchos factores que se escapan del control de los mismos, relacionados con la anatomía y el comportamiento del sistema cardiovascular. En el peor de los casos, uno de estos accidentes puede ocasionar la muerte al paciente, sin embargo en la mayoría de los casos éstos sufren parálisis en sus miembros y ven afectada su capacidad para hablar.

Cuando se tiene el infortunio de sufrir un ACV, se llevan a cabo valoraciones médicas de los afectados para determinar la gravedad de su lesión, y a partir de entonces, se procede a rehabilitarlos por medio de terapias físicas, las cuales se adecúan a la patología del paciente y duran el tiempo que le tome al cuerpo recuperarse.

El presente proyecto de investigación centra su atención en los problemas de movilidad en las manos. En estos casos, las terapias de rehabilitación se enfocan en desarrollar ejercicios de flexión y extensión de los dedos, y de rotación de la muñeca, a fin de enseñarle de nuevo a la mano a realizar sus movimientos más básicos [1]. En los casos en que las lesiones son muy graves, es posible que el paciente no recupere del todo la movilidad de sus manos o que simplemente tome mucho tiempo para ello, así las terapias se convierten en tareas repetitivas, en las cuales la amplitud de los movimientos realizados y la presión ejercida sobre los músculos afectados se incrementa al ritmo que el paciente se recupere [2].

Esta tendencia a repetir ejercicios y modificarlos lentamente da cabida a la utilización de robots para asistir las terapias, ya que se les puede enseñar los movimientos con que deben ayudar a los pacientes y cambiar de una sesión a otra los límites impuestos para dichos movimientos. Entonces el robot le ahorraría al fisioterapeuta la labor de realizar estos ejercicios repetitivos e incluso acelerarían la recuperación del paciente si éste pudiera utilizar el robot en su casa, ya que podría ejercitar más seguido sus músculos y evitaría invertir tiempo en movilizarse hasta el centro de rehabilitación. Además, los sistemas robóticos de rehabilitación actuales incluyen aplicaciones software que permiten programar los ejercicios que se llevan a cabo en las terapias y motivan a los pacientes para realizarlos, lo cual hace muy agradable la experiencia de utilizar sistemas de este tipo.

Por esta razón, en el presente trabajo se plantean como objetivos diseñar un sistema de rehabilitación para mano derecha, desarrollar una interfaz gráfica para programar las terapias, y por supuesto implementar el sistema diseñado. Dicho sistema tiene la particularidad de que utiliza el Nitinol como sistema actuador. Además se consideró también el alto costo que tienen los equipos de rehabilitación disponibles en el mercado, por lo cual se utilizan herramientas de software libre y materiales de construcción económicos, tal que el costo del dispositivo final estuviera al alcance de gran parte de la población interesada.

Este documento se organiza de la siguiente manera. En el capítulo 2 se presenta un estudio de las características morfológicas de la mano. En el capítulo 3 se describen las SMA (*Shape Memory Alloy*) y en particular se centra la atención en el Nitinol como

sistema de actuación, destacando las ventajas que representa el uso de estos materiales frente a otras alternativas como son los motores o los sistemas neumáticos de actuación. En el capítulo 4 se presentan los experimentos realizados con esta aleación y los resultados obtenidos. En el capítulo 5 se realiza una descripción de las herramientas software utilizadas para desarrollar la aplicación del proyecto. En el capítulo 6 se presenta el diseño propuesto para el robot, su proceso de construcción y cómo se desarrolló la aplicación software. En el capítulo 7 se exponen las sesiones de rehabilitación experimentales que se llevaron a cabo para observar el comportamiento del dispositivo diseñado y se proponen algunas sugerencias para mejorar el diseño en futuras versiones. El capítulo 8 se dedica a las conclusiones referentes al trabajo realizado y en el capítulo 9 se muestran las referencias empleadas.

Para la culminación exitosa del proyecto, se contó con el invaluable apoyo académico y material del PhD. José María Sabater Navarro del Departamento de Ingeniería de Sistemas y Automática de la Universidad Miguel Hernández de Elche, donde se tuvo la oportunidad de realizar una pasantía de 3 meses para llevar a cabo la implementación del robot. Dicha pasantía fue soportada económicamente por el proyecto OpenSurg, dirigido por el Dr. Sabater, proyecto financiado por CYTED. También se agradece el asesoramiento médico por parte del Departamento de Fisioterapia de la Universidad del Cauca a cargo de su directora, la fisioterapeuta María Verónica Torres, y en particular las sugerencias para el diseño de la interfaz de usuario y la supervisión durante las sesiones experimentales por parte del fisioterapeuta Jerónimo Londoño.

2. La mano humana

La mano humana es la herramienta más versátil y práctica que posee el hombre, con la cual experimenta sensaciones, tiene contacto con el entorno, y elabora otras herramientas para aumentar sus capacidades y realizar toda clase de trabajos. Para conseguir toda esta funcionalidad, la mano cuenta con un sistema muy desarrollado de músculos y tendones que le dan la posibilidad de lograr gran variedad de agarres y realizar tanto las tareas que implican fuerza bruta como aquellas que requieren precisión y destreza.

2.1. Características, funcionalidad y agarres

La mano está compuesta por la palma y los dedos, siendo la muñeca el punto de unión de la mano con el brazo. La mano tiene 27 huesos que se agrupan como carpio, metacarpo y falanges (Figura 1 (a)). Cuenta con 40 músculos agrupados como intrínsecos si se originan e insertan dentro de la mano, y extrínsecos si se originan fuera de la mano y se insertan en ella. Esta estructura dota de 20 grados de libertad a la mano humana, sin tener en cuenta los movimientos de la muñeca [1].

Las articulaciones son las que permiten los movimientos, los cuales dependen de la forma que tengan las superficies de contacto de los huesos. El nombre que recibe cada articulación se da según los huesos que estén unidos, tal como se muestra en la Figura 1 (b). Así el pulgar tiene, desde la muñeca hacia el extremo distal, las articulaciones carpometacarpiana, metacarpofalángica e interfalángica, y los otros 4 dedos tienen, desde la palma hacia el extremo distal, las articulaciones metacarpofalángicas, interfalángicas proximales e interfalángicas distales. Así mismo, por convención se acostumbra enumerar los dedos, siendo el pulgar el número 1, el índice el 2, el medio el 3, el anular el 4 y el meñique el 5 [3].

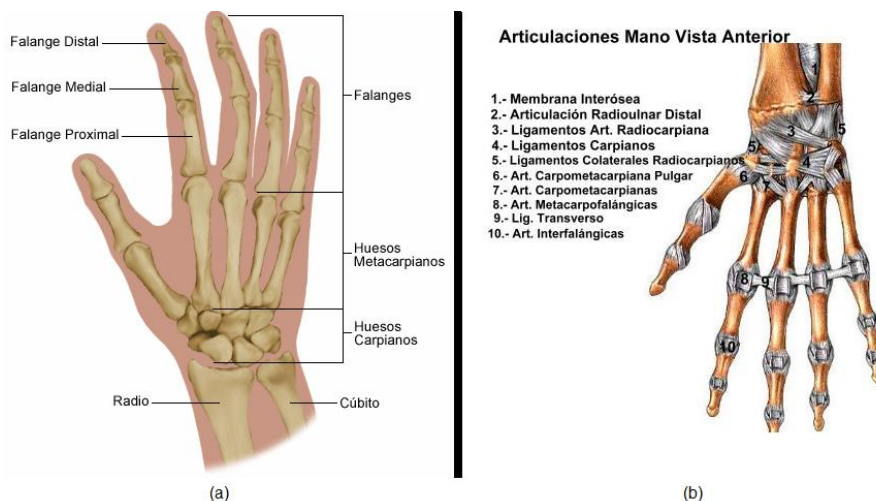


Figura 1. Estructura ósea de la mano.
Fuente: [4]

Los tendones son la conexión entre los músculos y la estructura ósea (Figura 2). Como están hechos de colágeno poseen la elasticidad necesaria para llevar a cabo la función de reposicionamiento de los huesos [4]. Muchos de ellos pasan a través de distintas

articulaciones antes de insertarse en un hueso, y según el movimiento que deban transmitir, pueden combinarse con otros tendones o bifurcarse antes de la inserción [1].

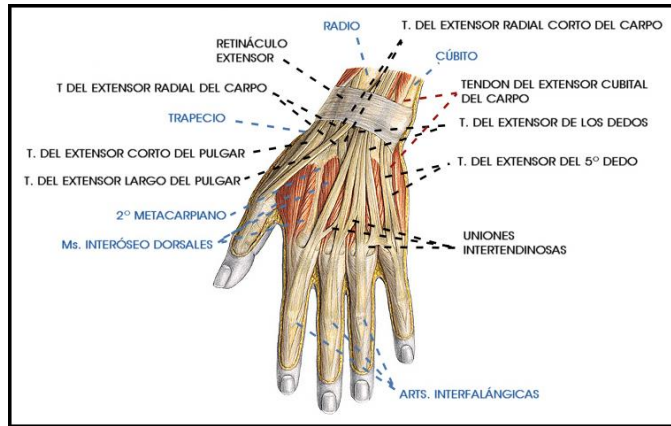


Figura 2. Tendones extensores de la mano.
Fuente: [5]

Sin embargo, los tendones no pueden accionar ninguna articulación, tan solo transmiten la fuerza generada por los músculos, quienes son considerados como un motor si se piensa en la mano como un mecanismo. De los 40 músculos que posee la mano, la muñeca cuenta con 5 músculos dedicados a su movimiento, el pulgar cuenta con 9, el índice y el meñique con 7 músculos, y hay 6 dedicados para el medio y el anular [1]. Es por esta razón que se pueden lograr desde movimientos precisos hasta agarres con mucha fuerza, de modo que aparte de clasificar los músculos como intrínsecos y extrínsecos, también se puede separarlos según el tipo de movimiento que permiten realizar, entonces pueden ser flexores, extensores, abductores o aductores.

Ahora bien, la funcionalidad de la mano se percibe en la gran cantidad de agarres que permite, los cuales se pueden apreciar en la Figura 3 en el siguiente orden: (A) cilíndrico, (B) de punta, (C) de gancho, (D) palmar, (E) esférico y (F) lateral [6].

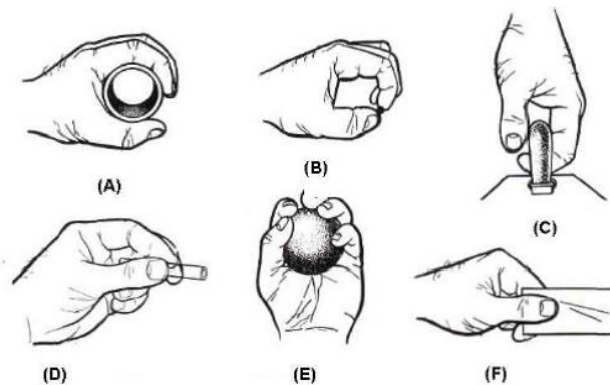


Figura 3. Agarres prensiles humanos.
Fuente: [6]

2.2. Accidentes cerebro-vasculares

Los accidentes cerebro-vasculares, conocidos también como derrames cerebrales, son ocasionados porque falla el suministro de sangre al cerebro ocasionando muchas veces la muerte o dejando daño cerebral permanente [3].

Según la causa de los ACV, es posible clasificarlos como isquemias cerebrales o como derrames hemorrágicos. Una isquemia cerebral es causada por una obstrucción arterial, normalmente a causa de un coágulo de sangre y puede ser de 2 tipos. Si el coágulo se forma en alguna parte del cuerpo y viaja por el torrente sanguíneo hasta el cerebro se conoce como embolia cerebral, y si el coágulo se forma dentro de un vaso en el cerebro se le llama trombosis. Los derrames hemorrágicos se dan cuando se rompe una arteria en el cerebro o en su superficie, lo cual es normalmente atribuido a una zona débil en la pared arterial, característica conocida como aneurisma [7].

Las discapacidades producidas por los ACV se pueden agrupar en 5 tipos según el efecto producido en el afectado [8]:

- **Parálisis o problemas con el control de los movimientos:** Es la secuela que padece la mayoría de los afectados por los ACV. Habitualmente esta parálisis se presenta en una mitad del cuerpo, aunque en ocasiones sólo se afectan algunos miembros, y se denomina hemiplejía si la parálisis es total o hemiparesia si el paciente conserva algo de movilidad en sus miembros afectados, claro que conservan esta movilidad con poco control y precisión.
- **Disminución en la sensibilidad:** Sucede cuando el daño al sistema nervioso afecta la capacidad del afectado para percibir sensaciones, tal como el contacto con objetos, cambios en la temperatura, o la percepción del dolor.
- **Problemas con el habla:** Sucede cuando se ve afectado el centro de control del lenguaje en el cerebro, lo cual se conoce como afasia. La afasia puede ser expresiva si los pacientes tienen problemas para construir frases coherentes o decir lo que están pensando, o puede ser receptiva si los pacientes presentan problemas para entender lo que les dicen o lo que leen. En los casos más serios, se puede dar afasia global, la cual combina la expresiva y la receptiva.
- **Problemas con el pensamiento y la memoria:** En estos casos los pacientes presentan dificultad para aprender nuevas tareas o para planear actividades, y se disminuye su capacidad para atender y la memoria de corto plazo.
- **Problemas emocionales:** A causa de los efectos físicos que producen los ACV, los pacientes sienten depresión, tristeza, miedo, ansiedad, frustración y enojo.

2.3. Terapias y sistemas de rehabilitación de mano

A pesar de los diferentes tipos de padecimientos que sufren los afectados por derrames cerebrales, la presente investigación se centra en las parálisis o problemas con el control de los movimientos, en concreto, en los problemas de movilidad en las manos.

En la rama de la medicina, los encargados de tratar estos problemas de movilidad son los fisioterapeutas. Estos profesionales ayudan a los pacientes a recuperar la fuerza, resistencia, rango de movimiento y sensibilidad en sus partes afectadas por medio de programas de rehabilitación pensados para cada paciente, según la patología que presente. La importancia de la labor de los fisioterapeutas radica en que los ejercicios que aplican en los pacientes no sólo ayudan a recuperar la fuerza en los miembros paralizados, sino que, al ser repetitivos, también sirven para estimular y mentalizar a los pacientes para que usen de nuevo sus miembros afectados tal como lo hacían antes del accidente, aprovechando la plasticidad del cerebro¹ [8].

Los ACV producen espasticidad² en los pacientes. El aumento en el tono muscular (estado de tensión y excitación de los músculos) se manifiesta más rápido en los músculos flexores que en los extensores, haciendo que los miembros afectados se mantengan recogidos, y en vista que se pierde control sobre la contracción voluntaria de los músculos, el paciente sufre disminuciones en el rango de movimiento pasivo y activo de sus articulaciones [3].

Entonces los ejercicios para rehabilitación en pacientes que presentan pérdida en la movilidad de sus manos se enfocan en disminuir la rigidez de los músculos que han aumentado su tono y aumentar gradualmente el rango de movimiento de las articulaciones mientras se fortalecen aquellos músculos que presentan debilidad. De esta manera también se ayuda a que el cerebro aprenda de nuevo a controlar los músculos y movimientos voluntarios, con lo cual se consigue la recuperación satisfactoria del paciente. Para lograr esta recuperación, las terapias de rehabilitación de mano persiguen los siguientes objetivos [2]:

- Agarrar objetos y controlar la posición de la muñeca durante el agarre.
- Liberar de manera voluntaria el agarre de los objetos.
- Lograr extensión activa con resistencia para cada dedo.
- Alternar entre extensión y flexión voluntaria de cada dedo individualmente.

Para conseguir estos objetivos se llevan a cabo terapias para el rango de movimiento, de coordinación mano-ojo y enfocada a las actividades cotidianas. Aquellas para el rango de movimiento pueden ser activas o pasivas y se basan en la apertura y cierre que pueden lograr las articulaciones, por lo cual se usa esta apertura permitida como indicador de la rehabilitación de los pacientes. Para esto puede tomarse como referencia la

¹ La neuroplasticidad es la capacidad del cerebro para asignar las funciones realizadas por alguna región dañada a otras regiones.

² La espasticidad se produce por daños en el sistema nervioso motor, y consiste en la contracción permanente de los músculos, haciéndolos rígidos y acortándolos.

Tabla 1. Rango de movimiento de las articulaciones de los dedos.
Fuente: [6]

Dedos	Articulación	Número de grados de libertad	Rango de movimiento	
			Rango de flexión-extensión	Rango de aducción-abducción
Meñique al índice	DIP	1	0° a 85°	----
	PIP	1	-10° a 120°	----
	MCP	2	-80° a 120°	0° a 60°
Pulgar	IP	1	-45° a 90°	----
	MCP	2	-45° a 90°	-15 ° a 15°
	CMC	2	-20° a 45°	-10° a 45°

Las terapias para coordinación mano-ojo consisten en que el paciente debe usar su miembro afectado para seguir el contorno de diferentes figuras geométricas u otros tipos de curva, de modo que se puede medir la recuperación del paciente por medio del error que hay entre la trayectoria descrita por su mano y la que debe seguir de acuerdo al ejercicio que se realice.

Por último, las terapias enfocadas a las actividades cotidianas consisten en que el paciente debe realizar tareas del hogar, tal como vestirse o comer, procurando utilizar su mano afectada en todo caso. En ocasiones se bloquea el miembro en buen estado para que el paciente haga todo lo posible por usar sólo su extremidad afectada, creando conciencia de cada movimiento, de cada orden que le envía a su mano y enfrentando el reto que implica contar sólo con su mano enferma para desarrollar las actividades de la terapia, así no solo se trabaja el aspecto físico del paciente sino también con el psicológico. No está de más mencionar que estas terapias son controladas por el fisioterapeuta, quien indica al paciente lo que puede ir haciendo en casa y supervisa y evalúa posteriormente su evolución [2].

La robótica ha encontrado un lugar en este campo de la rehabilitación, ya que se han desarrollado robots asistentes para el desarrollo de las terapias físicas. Entonces, los robots pueden realizar con los pacientes los ejercicios de movimiento pasivo asistido de modo que los fisioterapeutas solamente supervisen los resultados, y también pueden aplicar carga en dedos y manos para llevar a cabo los ejercicios para movimientos activos. Adicionalmente, es posible desarrollar los robots con juegos o aplicaciones software enfocadas a que los pacientes realicen ejercicios de coordinación mano-ojo, haciendo que la utilización del robot y la ejecución de la terapia sea una experiencia agradable y enriquecedora.

Sin embargo, es necesario mencionar que no existe como tal una rutina de ejercicios que se ejecuten en las terapias de rehabilitación de manos. En cada terapia el fisioterapeuta hace un diagnóstico del grado de recuperación del paciente, evaluando el rango de movimiento permitido por cada articulación y realiza algunos ejercicios a manera de calentamiento para acondicionar al paciente para la terapia, los cuales son completamente pasivos. Las terapias varían entonces según el estado del paciente y el criterio del fisioterapeuta, por eso no se pueden programar rutinas en un dispositivo para rehabilitación de manos. En cambio sí es posible programar ejercicios según el tipo de

movimiento que se desea generar y luego alternarlos de acuerdo a cada paciente, por ejemplo, programar un ejercicio para flexión, otro para extensión, otro para agarre cilíndrico, otro para agarre tipo punta, etc., de tal manera que se cubran las funciones que puede perder la mano tras el ACV.

En vista que la mayoría de los pacientes quedan con sus manos fuertemente cerradas después del derrame, los fisioterapeutas buscan primero liberar el pulgar de los otros dedos, para así proceder a la rehabilitación de los 5 dedos por igual. Aparte de esto, como recomendación general, estos ejercicios se realizan bloqueando el movimiento de la muñeca, de modo que sólo sea posible mover las articulaciones de los dedos, y en vista que los músculos más afectados son los extensores, se trata de evitar el contacto con la palma de la mano cuando se realizan ejercicios de extensión, ya que ese contacto con la palma genera una reacción refleja de cierre de la mano.

2.4. Sistemas robóticos y mecánicos para rehabilitación de mano

- **Hexxor:** Es un sistema diseñado para la rehabilitación de mano en pacientes con hemiparesia. Se busca que el paciente logre incrementar el rango de movimiento, la fuerza en el agarre y el control preciso de su mano por medio de los ejercicios que realiza con el robot. En cada terapia se registra el progreso de los pacientes con la ayuda de los sensores de fuerza y posición montados en el sistema. El sistema cuenta con un mecanismo de 4 barras de tipo manivela-balancín para accionar los dedos, destinando el eslabón de entrada para mover las articulaciones metacarpofalángicas y el acoplador para mover las proximales interfalángicas. Las articulaciones distales interfalángicas no son accionadas directamente [3]. HEXXOR se muestra en la Figura 4.

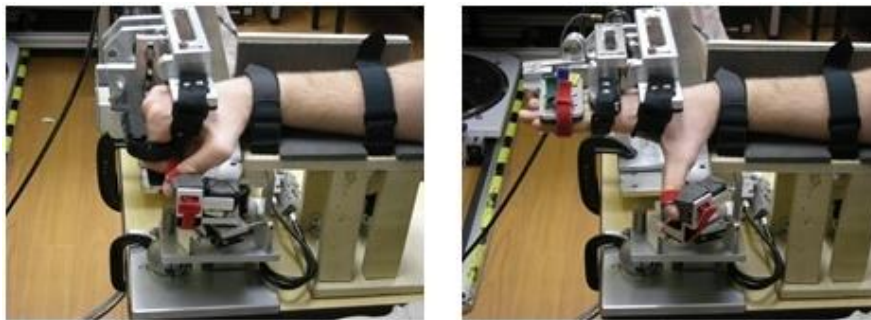


Figura 4. HEXXOR.
Fuente: [9]

- **FTM:** Sigla de "*Functional Tone Management*" es un programa de entrenamiento para los brazos de personas que han sufrido accidentes cerebro-vasculares y presentan hemiparesia. No posee sistemas electrónicos, consiste en un guante que se acopla a la mano y antebrazo, y por medio de resortes, ayuda al paciente a recuperar la fuerza y la movilidad de sus dedos. El guante mantiene la mano en una postura adecuada para llevar a cabo las tareas de agarre y liberación del agarre, y se puede poner fácilmente con una sola mano, de modo que se presta para que el paciente la utilice en casa ya que el paquete incluye también un programa de ejercicios y algunos accesorios para poder realizarlos, tal como pelotas y aros de plástico [10].

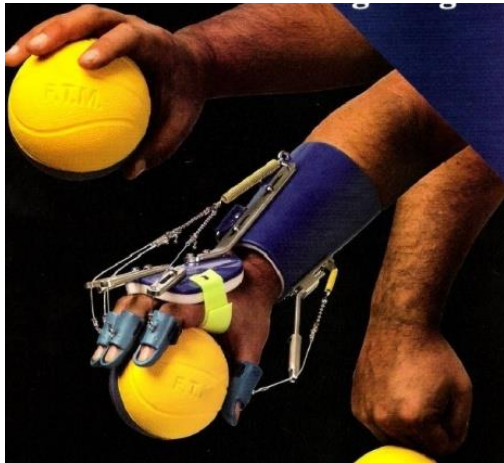


Figura 5. FTM rehabilitation device.
Fuente: [1]

- Hand Mentor:** Es un dispositivo diseñado para rehabilitación de muñeca y dedos. Fue nombrado “mentor” porque da instrucciones al paciente durante las terapias. Permite realizar ejercicios para movimiento activo y pasivo, para lo cual cuenta con un músculo neumático que asiste los movimientos que lo requieran. Las terapias se desarrollan por medio de juegos, los cuales están orientados para 2 clases de pacientes. Aquellos con un tono muscular elevado cuentan con el programa de reducción de espasticidad, mejorando el rango de movimiento pasivo, y aquellos con capacidad para realizar algunos movimientos por sí mismos, pueden emplear el programa de control motor para mejorar el rango de movimiento activo. En la interfaz del dispositivo se muestra también información sobre la fuerza, posición y la actividad eléctrica de los músculos del paciente, la cual es almacenada junto a los otros datos de la terapia en la historia del usuario [11]. El robot se muestra en la Figura 6.



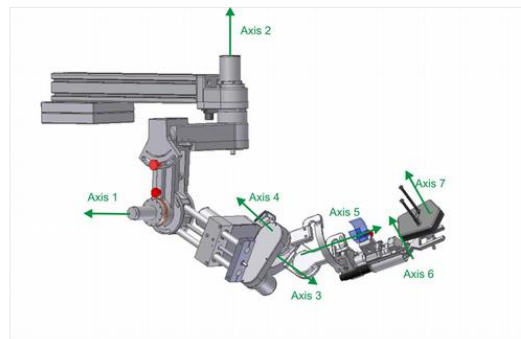
Figura 6. The Hand Mentor.
Fuente: [11]

- ARMin:** Desarrollado en la Universidad de Zurich para rehabilitación de brazo completo (Figura 7 (a)). Cuenta con 7 grados de libertad activos (Figura 7 (b)), y se ajusta a la altura y dimensiones del brazo del paciente. Ofrece 3 modos de entrenamiento, el primero es llamado modo de movilización, en el cual el

fisioterapeuta le enseña al robot el movimiento que debe realizar, éste graba la trayectoria, la suaviza, y aplica el movimiento al brazo del paciente, es un modo totalmente pasivo y se usa a manera de calentamiento, ya que el movimiento pasivo asistido relaja al paciente, reduciendo el efecto de la espasticidad y estimulando la circulación de la sangre. El segundo modo es el de juegos, los cuales trabajan bien sea, una sola articulación, grupos de ellas, o todas. Y el tercer modo es el de actividades de la vida cotidiana, en el cual se simulan entornos habituales de la casa y la calle con el fin que el paciente realice tareas como utilizar la estufa o comprar bebidas en una máquina expendedora, pero con la asistencia del robot [12].



(a)



(b)

Figura 7. ARMin III.

Fuente: [12]

- **Amadeo:** Este sistema tiene una ventaja sobre los demás dispositivos robóticos para rehabilitación que consiste en que permite mover cada dedo de la mano de manera individual, y por supuesto también permite mover todos juntos, bien sea en una secuencia programada o de manera aleatoria. Para lograr esta independencia en los movimientos, el sistema cuenta con bielas para cada dedo, las cuales unen las puntas de los dedos con el sistema de actuación. El software del dispositivo permite trabajar en modo de terapia pasiva, movimientos asistidos y terapia activa e interactiva en caso de usar los juegos. Permite la medición del rango de movimiento y de la fuerza aplicada por cada dedo, lo cual se muestra al paciente por medio de la interfaz de usuario al mismo tiempo que se almacena en su historia para posteriores evaluaciones del desempeño [13]. Amadeo se muestra en la Figura 8.



Figura 8. Amadeo: Finger and Hand Rehabilitation.
Fuente: [13]

Del estudio de estos dispositivos para rehabilitación de mano se puede observar que en todos los casos, se necesita que el paciente pueda abrir su mano para ponerse o utilizar los dispositivos. Entonces, cuando los pacientes presentan alta espasticidad después del accidente, deben ser los fisioterapeutas quienes realicen la primera parte de la rehabilitación sin la asistencia de ningún robot hasta que se logre una reducción en el tono muscular tal que el paciente no ofrezca mucha resistencia ni experimente dolor al abrir su mano para que a partir de ese momento pueda usar alguno de los dispositivos para rehabilitación.

En todo caso, si se desea construir un dispositivo que efectivamente sirva en la mayoría de etapas de recuperación de un ACV, éste debe ser capaz de adaptarse a la condición del paciente bien sea que presente hipertonía o hipotonía³, es decir que el sistema de agarre de los dedos debe poder ajustarse a una mano flácida y débil o a una mano cerrada y tensionada. Sin embargo tal vez no sea necesario utilizar uno de estos dispositivos en la segunda situación ya que en estos casos la terapia se enfoca normalmente a relajar los músculos, reducir la hipertonía, y reducir el dolor experimentado al mover los dedos o la mano, lo cual no se puede lograr con una máquina orientada a realizar ejercicios repetitivos.

Entonces, el diseño de un dispositivo para rehabilitación de mano debe facilitar su utilización por parte del paciente. Así debe buscarse la manera para que el montaje de la mano sobre el sistema sea rápido y sencillo. Por ejemplo, no es muy práctico diseñar un dispositivo con forma de guante ya que sería muy difícil que un paciente espástico se lo colocara.

³ La hipertonía es el aumento del tono muscular, con aumento de la resistencia al estiramiento pasivo. La hipotonía es el opuesto a la hipertonía, y consiste en la disminución del tono muscular, haciendo a los músculos flácidos y débiles.

3. Aleaciones con memoria de forma

En los metales tradicionales, las deformaciones hacen que la estructura molecular se deshaga, dejando a los átomos en nuevas posiciones cristalinas, a causa de esto el cristal no puede conservar una “memoria” de dónde estaban los átomos antes de moverse [14]. Además otro comportamiento común es que al someter los metales a altas temperaturas estos tienden a expandirse y a bajas temperaturas se contraen. Dichas características no son propias de un SMA siendo esta una aleación compuesta por dos o más metales, lo que las hace bastante peculiares.

Las aleaciones SMA son aleaciones metálicas que después de una deformación aparentemente plástica, vuelven a su forma original cuando son sometidas a un calentamiento [15]. A esto se le conoce como efecto de memoria de forma. Los mismos materiales dentro de un determinado rango de temperaturas pueden ser deformados hasta casi un 10% de su longitud, aunque algunas aleaciones en forma de muelle pueden llegar a deformarse hasta 0.14 m de longitud y volver a recuperar su forma original al ser calentados [16]. Estos efectos son llamados memoria de forma térmica y memoria de forma elástica. Ambos efectos se deben a un cambio de fase llamado transformación martensítica termo elástica. Las aplicaciones potenciales de estos dos principales comportamientos permiten generar fuerza, movimiento, o almacenar energía [17].

Fases de transformación.

El estado por el cual las SMA recuperan su forma es el resultado de una transformación sólida-sólida entre dos estructuras de materiales conocidas como fases austenita y martensita. Existe una histéresis asociada a esta transformación y un rango de temperatura en que coexisten austenita y martensita [17]. La transformación de la fase martensita puede ser caracterizada por las temperaturas de comienzo (A_s) y fin (A_f) de fase austenita, y las temperaturas de comienzo (M_s) y fin (M_f) de fase martensita [18].

Desde el punto de vista macro el efecto de memoria de forma puede describirse como se muestra en la Figura 9. No existe cambio en la forma si la temperatura se encuentra por encima de A_f o se encuentra en la fase austenita manteniendo la forma original. Cuando el elemento es deformado por debajo de M_f permanece con esa deformación hasta que se calienta. La recuperación de la forma comienza en A_s y es completada en A_f . En el punto de inflexión entre A_s y A_f , cerca del 50% de la forma original ya está recuperada. Una vez que la forma se ha recuperado en A_f no hay más cambios en la forma cuando el elemento es enfriado hasta por debajo de M_f , y la memoria de forma puede solo ser reactivada deformando la martensita otra vez. En otras palabras el efecto de memoria de forma ocurre solo una vez, y por eso se suele llamar memoria de forma simple, en contraste con el efecto de memoria doble que presentan algunas aleaciones. Las deformaciones de recuperación son del orden del 7 % en las aleaciones SMA, aunque algunas pueden llegar al 10%. Los procesos de transformación de austenita a martensita y viceversa siguen diferentes caminos, como consecuencia se genera histéresis en el proceso de transformación [17].

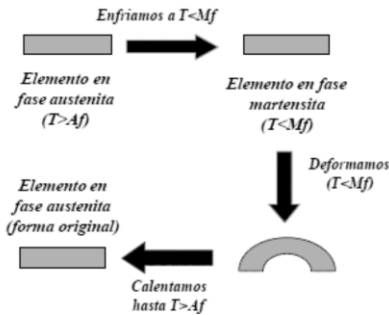


Figura 9. Representación macro de la transformación de fases
Fuente: [17].

Efecto de memoria simple y doble.

En la Figura 10 se muestra una comparación entre los tipos de memoria simple y doble. En la primera la aleación solo recuerda o recupera la forma definida en la fase austenita, por lo que es necesario aplicar fuerza a una temperatura baja si se desea deformar nuevamente la aleación.

En una aleación con memoria de forma doble, el material tiene la habilidad de recordar o recuperar las formas definidas en las dos fases de transformación.

En ambos casos solo es posible generar trabajo durante la fase calentamiento [19].

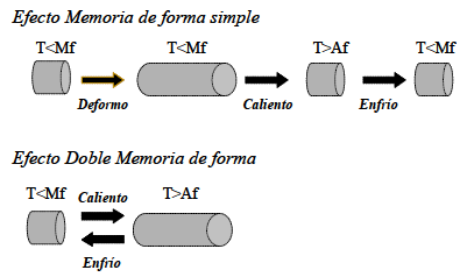


Figura 10. Efecto simple y doble de memoria de forma
Fuente: [17].

3.1. Tipos de SMA

Se puede obtener una gran variedad de aleaciones, con solo variar los porcentajes de concentración de los metales que componen la aleación o con sustituirlos por otros componentes.

Por ejemplo, las aleaciones en base a cobre son una opción más económica con respecto a las aleaciones de Ni-Ti (Níquel-Titanio), además de tener un procedimiento de fabricación más simple. Pero presentan algunas desventajas, como su baja ductibilidad⁴ y resultan ser más propensas a la corrosión lo que genera un rápido envejecimiento. Otro tipo de aleación como la de Cu-Al (Cobre-Aluminio) posee una temperatura de transformación muy alta.

⁴ propiedad de los metales para formar alambres o hilos.

Existen también aleaciones que incorporan un tercer o cuarto elemento con el propósito de modificar la temperatura de transformación. Las aleaciones más estudiadas de este tipo son la Cu-Zn-Al (Cobre-Zinc-Aluminio) y Cu-Al-Ni (Cobre-Aluminio-Niquel), disponibles comercialmente. Al agregar Ni, se obtiene un excelente efecto de memoria de forma, además de una mejor estabilidad térmica y mayores temperaturas de operación que la aleación Cu-Zn-Al. La desventaja que presenta es que al aumentar el contenido de Ni, la aleación se vuelve frágil [20].

Entre los SMA más difundidas comercialmente encontramos las de NiTi, NiTi-X (X representa un elemento ternario) y Cu-Zn-Al. En la Actualidad, el 90 % de las nuevas aplicaciones están basadas en NiTi, NiTiCu y NiTiNb (Niquel-Titanio-Niobio). Otras aleaciones como CuAlNi o FeMnSi (Hierro-Manganeso-Silicio) comienzan a introducirse en el mercado y otras son potencialmente interesantes como las de NiAl o NiTiZr (Niquel-Titanio-Circonio) pero son excesivamente frágiles. También se están centrando las investigaciones en aleaciones con Pt (platino) como elemento base por dos motivos fundamentales: por un lado las aleaciones de NiTi y CuZnAl solo pueden ser usadas hasta alrededor de los 100°C, con lo que se requieren de aleaciones de utilización en mayores temperaturas y por otro lado, porque ni Cu ni el Ni son biocompatibles y una de las aplicaciones más interesantes de las SMA es en implantes.

Como comparación, se presenta la composición química y el rango de temperaturas de las principales aleaciones SMA.

Tabla 2. Composición química y propiedades de algunas aleaciones SMA
Fuente [21].

Aleación	Composición	Rango de temperaturas (°C)	Histéresis (°C)
Ag-Cd	44/49 at %Cd	-190 a -50	15
Au-Cd	46.5/50 at Cd	30 a 100	15
Cu-Al-Ni	14/14.5 at %Al; 3/4.5 wt% Ni	-140 a 100	35
Cu-Sn	15 at % Sn	-120 a 30	
Cu-Zn	38.5/41.5 wt %Zn	-180 a -10	10
In-Ti	18/23 at % Ti	60 a100	4
Ni-al	36/38 at % Ti	-180 a 100	10
Ni-Ti	49/51 at % Ni	-50 a 110	30
Fe-Pt	25 at % Pt	-130	40
Mn-Cu	5/35 at % Cu	-250 a 180	24
Fe-Mn-Si	32 wt%Mn; 6wt%Si	-200 a 150	100

3.1.1. El Nitinol

La primera aleación con memoria de forma en el ámbito de la ingeniería ha sido la aleación Ni-Ti o Nitinol. Esta fue descubierta en 1958 por William Buehler en uno de los laboratorios Navales de los Estados Unidos, de ahí su nombre Nitinol (*Nickel Titanium Naval Ordnance Laboratory*). El problema de esta aleación es su alto costo, principalmente asociado a la producción de ésta.

Pese a todas las investigaciones en busca de nuevas aleaciones con memoria de forma, las únicas aleaciones comerciales en la actualidad son las NiTi y las bases de Cu, donde

predominan las primeras. Esto sucede por múltiples razones: las aleaciones NiTi tienen mayor capacidad de memoria (hasta un 8% mientras que solo se alcanza un 4-5% en las base Cu), además son mucho más estables térmicamente, tienen una excelente resistencia a la corrosión comparadas con las de cobre, tienen mayor resistencia eléctrica (siendo su activación eléctrica más simple), pueden ser aleadas⁵ y extruidas⁶ con facilidad y tienen un mayor rango de posibles temperaturas de transformación. Además, en aplicaciones médicas, el Cu está prácticamente eliminado. Por otro lado, las aleaciones base cobre (básicamente CuZnAl y CuAlNi) pueden sufrir fallos intergranulares debido a que su estructura granular intrínseca es muy basta y, pese a que se han ido desarrollando muchas técnicas para refinar la estructura granular, sus propiedades siguen siendo peores comparadas con las aleaciones NiTi [17].

Estas ventajas sumadas al hecho de que comercialmente son relativamente más fáciles de adquirir motivaron su elección para ser empleadas como actuadores de nuestro sistema de rehabilitación.

Como gran inconveniente de las aleaciones NiTi, podemos mencionar que son mucho más caras y difíciles de mecanizar. Además, el ciclo de histéresis de las aleaciones NiTi suele ser muy pronunciado lo que origina grandes cambios en la deformación con pequeños cambios de temperatura dificultando su control [17].

3.1.2. Propiedades de la aleación NiTi

En la Tabla 3 se presentan las características físicas mecánicas y de forma de una aleación Niti en particular.

Tabla 3. Propiedades físicas, mecánicas y de forma de una aleación NiTi
Fuente [22].

PROPIEDADES FISICAS		
Punto de fusión	2390°F	1310°C
Densidad	0.234 lb/in ³	6.5 g/cm ³
Resistividad eléctrica	30 μohm-in	76 μohm-cm
Módulo de elasticidad	4-6 x 10 ⁶ psi	28-41 x 10 ³ MPa
Coefficiente de expansión térmica	3.7 x 10 ⁻⁶ /°F	6.6 x 10 ⁻⁶ /°C
PROPIEDADES MECANICAS		
Resistencia a la tracción	160 x 10 ³ psi	1100 MPa
Elongación total (min)	10%	10%
PROPIEDADES DE MEMORIA DE FORMA		
Deformación máxima (max)	8.0%	8.0%
Temperatura de transformación (Af)	140° F	60° C

⁵ Proceso por el cual se mezclan dos o más elementos de los cuales al menos uno es un metal.

⁶ Proceso por el cual se crean elementos con sección transversal definida.

En las aleaciones NiTi es posible modificar sustancialmente las temperaturas de transformación mediante pequeñas variaciones en el contenido de níquel y titanio, o bien sustituyendo níquel por cobalto. Es posible bajar la temperatura de transformación de fase aumentando el contenido de níquel, al tiempo que aumentamos la resistencia a la cedencia⁷ de la austenita. Sin embargo, si se añade níquel por encima del 55,6%, aparece una segunda fase estable (Ni₃-Ti) y las propiedades del NiTi se pierden. Por ello, para evitarlo, se suelen sustituir el níquel por el cobalto para bajar las temperaturas.

Otro de los comportamientos que se puede presentar en las aleaciones de NiTi es la aparición de una transformación de fase intermedia denominada transformación de fase R. Esta transformación precede a la transformación martensítica y es típica en aleaciones termomecánicamente tratadas ricas en Ni, en aleaciones envejecidas a una apropiada baja temperatura y en aleaciones ternarias como las NiTiFe y NiTiAl [23].

Dado que es una transformación de fase termo elástica, también puede dar lugar a un efecto de memoria de forma y a una superelasticidad, sin embargo, la amplitud de su histéresis es mucho más pequeña comparada con la transformación martensítica.

La transición de fase martensita genera una mayor fuerza [24], permitiendo poder emplear los SMA como actuadores.

La transición entre austenita y martensita es súper elástica, cualidad que permite poder deformar fácilmente los SMA [24].

3.2. Los SMA como actuadores

Un SMA puede funcionar como actuador realizando un trabajo contrario a una fuerza. Al calentarse usa el efecto de memoria de forma para generar fuerza y movimiento. Esta característica permite poder emplear el Nitinol como sistema de actuación final.

Se pueden mencionar las ventajas de emplear SMA como actuadores:

- Son sistemas silenciosos: no contienen componentes que produzcan o generen ruido.
- Diseño simple: no requieren componentes mecánicos para su funcionamiento.
- Tamaño y peso reducido: se pueden obtener aleaciones del orden de décimas de milímetro de grosor y cuyo peso no supera el gramo [25] .
- Biocompatibles: pueden ser empleados en biomedicina.
- Fiabilidad: si se da un uso adecuado al material se pueden obtener duraciones cercanas al millón de ciclos.
- Resistencia a la corrosión: existen aleaciones como NiTi que presentan alto grado de resistencia a la corrosión en comparación a aleaciones a base de Cu.

⁷ Se define como el punto en el cual, el material sufre deformación plástica, es decir el material se deforma permanentemente.

Ahora las desventajas de emplear SMA son:

- Baja eficiencia energética: la máxima eficiencia teórica de los SMA está en torno al 10%. Sin embargo la realidad es que la eficiencia es menor del 1% por lo que el SMA puede ser considerado un motor de calor operando a bajas temperaturas. La mayoría de la energía calorífica es perdida con el entorno. Por eso los actuadores SMA deben estar limitados a áreas donde la eficiencia energética no sea un requerimiento primordial [26].
- Degradación y fatiga: El rendimiento a lo largo del tiempo y la fiabilidad, dependen de numerosos factores incluyendo la máxima temperatura que se alcance, el estrés, la tensión y el número de ciclos de trabajado. Se tiene que tener un especial cuidado con los sobrecalentamientos y las sobretensiones en el actuador para que este dure el máximo tiempo posible. Sin embargo el desarrollo y la investigación con estos materiales están consiguiendo mejorar sus características [26].
- Baja velocidad: generalmente los actuadores de SMA son considerados lentos en cuanto a respuesta se refiere, debido a sus restricciones en enfriamiento y calefacción, así como a su ciclo térmico de histéresis. La manera tradicional de activar los SMA es a través de una corriente eléctrica, esto puede aumentar su velocidad de respuesta, pero los lleva a padecer mayor sobrecalentamiento [26].
- Control impreciso: El largo ciclo de histéresis, así como sus características no lineales en sus fases de transformación, hacen difícil controlar de manera precisa los SMA. Muchas investigaciones en control con este tipo de material han encontrado bajas velocidades de muestreo de tan solo 1Hz. Los tiempos de respuesta para una entrada escalón fueron de más de 1 segundo, y las precisiones fueron mediocres con errores de más de 1% del rango de trabajo [26].

Ahora dependiendo del trabajo final que realicen los SMA, se pueden clasificar en dos categorías: trabajo lineal o de revolución. En la Figura 11 se presentan aplicaciones de SMA como actuadores lineales o de revolución [26].

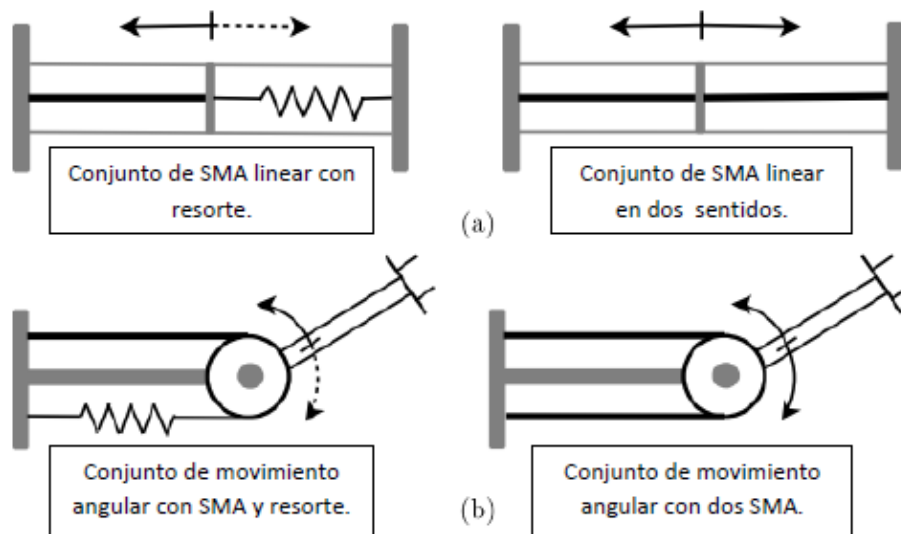


Figura 11. Configuraciones de empleo de SMA como actuador
Fuente [26].

4. Experimentación con Nitinol

En el capítulo anterior se justificó porqué se decide emplear Nitinol, ahora se determinará que características debe tener este para poder cumplir con los requerimientos de fuerza que requiere el sistema de rehabilitación.

En el trabajo “Estudio e implementación de actuadores basados en aleaciones SMA” se ha establecido que la fuerza mínima necesaria para lograr extender un dedo es de 15,9 N, siendo ésta la máxima fuerza generada por una articulación en la mano Shadow Dextrous, cuyo comportamiento busca imitar fielmente las características de una mano humana, incluyendo sus movimientos y fuerzas.

Los alambres y muelles de Nitinol de tensión con los que se experimenta fueron adquiridos al proveedor MuscleWiress [27].

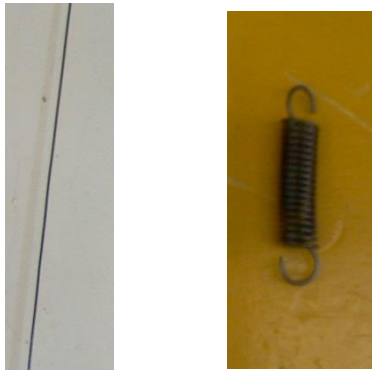


Figura 12. SMA de Nitinol empleados.

Las características físicas, eléctricas y térmicas se presentan a continuación:

Características físicas:

- Densidad: 0,235 lb/in³(6,45 g/cm³)
- Calor específico: 0,2 cal/g*°C
- Punto de fusión: 1300 °C
- Transformación latente de calor: 5,78 cal/g
- Conductividad térmica: 0,18 W/cm*°C
- Coeficiente de expansión térmica:
 - Fase Martensítica: 6,6x10⁻⁶/°C
 - Fase Austenita: 11,0x10⁻⁶/°C
- Resistividad Eléctrica:
 - Fase Martensítica: 80 micro ohmios*cm
 - Fase Austenita: 100 micro ohmios*cm

Características eléctricas:

Tabla 4. Características eléctricas Nitinol.
Fuente [28].

Diámetro (mm)	Fuerza generada (gm)	Fuerza generada (N)	Tiempo de refrigeración ⁸		Ciclo de trabajo ciclos/min		Corriente para 1 segundo de contracción (mA)
			70 °C "LT" Segundos	90 °C "HT" (Segundos)	LT ⁹	HT	
0,025	8,9	0,087	0,18	0,15	52	n.a	45
0,038	20	0,196	0,24	0,20	48	55	55
0,050	36	0,353	0,4	0,3	46	55	85
0,075	80	0,784	0,8	0,7	40	50	150
0,10	143	1,402	1,1	0,9	30	43	200
0,13	223	2,186	1,6	1,4	23	32	320
0,15	321	3,147	2,0	1,7	20	27	410
0,20	570	5,589	3,2	2,7	13	19	660
0,25	891	8,737	5,4	4,5	9	13	1050
0,31	1280	12,552	8,1	6,8	7	9	1500
0,38	2003	19,642	10,5	8,8	4	5	2250
0,51	3560	34,911	16,8	14,0			4000

Especificaciones térmicas:

Tabla 5. Especificaciones térmicas aleación Ni-Ti.
Fuente [28].

Parámetro	Aleación LT	Aleación HT
Temperatura inicial de activación (° C)	68	88
Temperatura final de activación (° C)	78	98
Temperatura inicial de relajación (° C)	52	72
Temperatura final de relajación (° C)	42	62
Temperatura de fusión (° C)	1300	1300
Calor específico (cal/g*° C)	0,077	0,077
Capacidad calorífica (Joule/g*° C)	0,32	0,32
Calor latente (Joule/g)	24.2	24,2

En los experimentos se emplea un calibre de alambre de 0,38 mm y muelles de Nitinol de 0,75 mm. El alambre es capaz de generar una fuerza de 19,642 N. En el caso del muelle, se debe considerar que asociado a este existe una constante de elasticidad que no es proporcionada por el fabricante, pero que se ha determinado experimentalmente en trabajos previos. En concreto se toma como referencia los resultados obtenidos en el trabajo de Taft y McAvoy [24], en el que empleando una variación del experimento clásico

⁸ El tiempo de refrigeración se ha realizado en una habitación con aire estático y empleando un cable en posición vertical. Este tiempo corresponde a un 95% de deformación, el último 5% no es tenido en cuenta.

⁹ LT y HT indican respectivamente que el material posee una baja y alta temperatura de activación

de física para obtener la constante de elasticidad de un resorte, determinan las constantes asociadas a las fases de transformación para un muelle de Nitinol. Este trabajo hace uso de un muelle de Nitinol que contiene las mismas propiedades que el empleado en este proyecto (Tabla 6). Como resultado se obtiene dos constantes de elasticidad, una asociada a cada fase de transformación, debido a que la transición entre las dos fases exhiben propiedades y comportamientos diferentes entre sí, se concluyó que la constante de fase austenita en la que se presenta una estructura cristalina cúbica, es la más alta con un valor de 220N/m a una temperatura máxima de 56 °C y, una constante de 183N/m asociada a la fase martensita en la que se obtiene una estructura cristalina deformada, a una temperatura de enfriamiento de 37 °C. Además este experimento logró establecer una temperatura de transición de 44 °C entre las dos fases.

Tabla 6. Propiedades físicas y térmicas muelle de Nitinol empleado.
Fuente [16].

Calibre	0,75 mm
Diámetro interno	3 mm
Deformación máxima	140 mm
Contracción mínima	29 mm
Temperatura de activación	45-55 °C
Corriente de Activación	4 A

En la Figura 13 se presenta el montaje de la maqueta empleada para realizar los experimentos con el alambre y el muelle de Nitinol respectivamente, con el fin de determinar la compresión-extensión máxima que pueden lograr y los tiempos empleados en retornar a la fase martensita, cuando se emplea refrigeración.

El experimento básicamente consiste en energizar un alambre o muelle de Nitinol que a su vez desplaza una varilla que se desliza a través de dos guías. El desplazamiento del Nitinol es medido con una regla puesta paralelamente al desplazamiento de este. Además se consignan los tiempos que tarda el Nitinol en retornar a su posición inicial empleando refrigeración generada por un ventilador de 12 Vdc.

Existen otras alternativas de refrigeración en las cuales se pueden obtener mejores resultados en los que el Nitinol tarde menos tiempo en volver a su fase inicial. Por ejemplo el fabricante especifica que se puede obtener mejores resultados si se sumerge el Nitinol en aceite o si se emplea como refrigerante agua con glicol [28], pero el uso de estos métodos implicaría incrementar la complejidad mecánica del sistema, ya que se requieren de elementos adicionales que se encarguen de manipular fluidos. Por tal motivo se opta por emplear ventiladores para generar aire forzado.

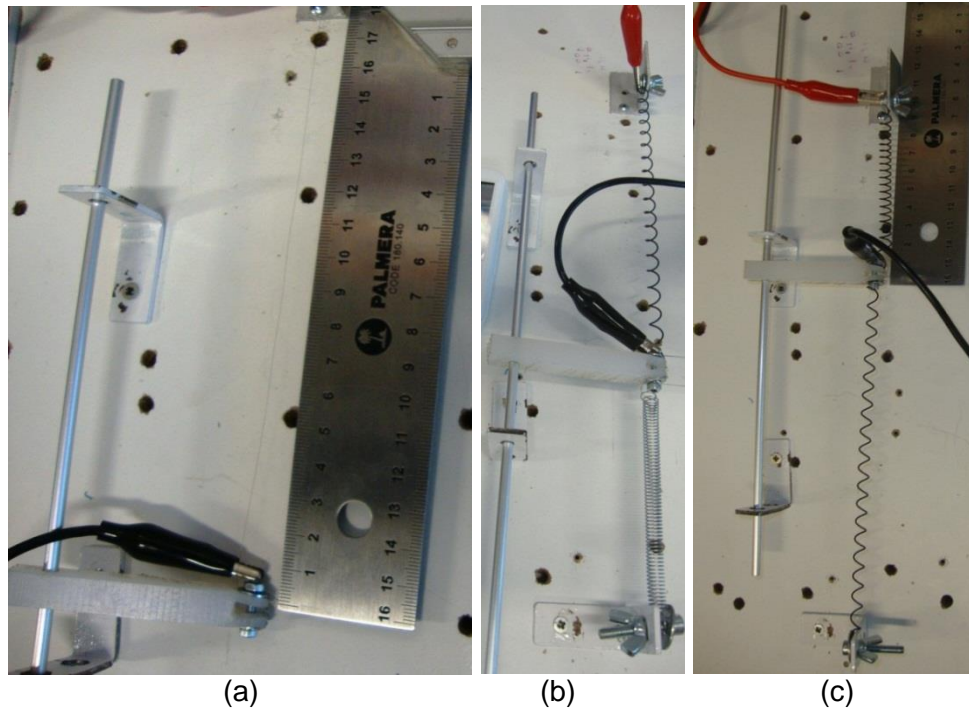


Figura 13. Maqueta de experimentación.

4.1. Experimento 1

En un experimento inicial, empleando la maqueta de la Figura 13(a), se emplea una longitud de 182 mm de alambre y se somete a una tensión de 3 V para proporcionar una corriente de 2,25 A, suficiente para lograr deformarlo. Los resultados se presentan en la Tabla 7.

Tabla 7. Resultados experimentales para un alambre de Nitinol de 0,38 mm.

Calibre (mm)	Longitud(mm)	Deformación(mm)	Corriente (A)	Tiempo 1	Tiempo 2
0,38	185	17	2,77	15,05	3,7

Las columnas Tiempo 1 y Tiempo 2 indican los tiempos promedio empleados en retornar el alambre a la posición inicial sin y con refrigeración respectivamente.

En la columna deformación se presentan las longitudes máximas deformadas al aplicar tensión al alambre de Nitinol. Si dividimos este valor entre la longitud empleada de alambre nos daremos cuenta que el porcentaje de deformación es de 9.18 % con respecto a su longitud total, por lo tanto este experimento permite concluir que para poder obtener una deformación de 11 cm, que es la longitud deseada para extender y flexionar los dedos sin el uso de un mecanismo que amplifique el recorrido, se requiere una longitud de alambre superior a los 130 cm.

4.2. Experimento 2

Evitar emplear mecanismos adicionales era un objetivo primordial ya que se deseaba un sistema de fácil fabricación, y que mecánicamente no fuera igual o superior en complejidad frente al uso de motores convencionales. Por tal motivo se descarta el uso de alambres y se opta por emplear muelles de Nitinol, ya que son capaces de soportar deformaciones por encima de los 10 cm, un valor muy superior comparado con las deformaciones obtenidas con el alambre de Nitinol. Como inconveniente estos muelles presentan características de memoria de forma simple, lo que obliga a emplear un mecanismo que permita retornar o deformar nuevamente el muelle para repetir el ciclo de funcionamiento.

Para solucionar este inconveniente se decide emplear resortes tradicionales (Figura 13 (b)) con una constante superior a 183N/m, que corresponde a la constante de elasticidad del muelle, para la fase martensita obtenida experimentalmente en el trabajo de Taft y McAvoy [24].

Emplear estos resortes permitió deformar nuevamente el resorte una longitud de 11cm pero incorporó un nuevo inconveniente. Ahora el muelle de Nitinol debe proporcionar una fuerza extra que deberá emplear para vencer la fuerza opuesta que genera el resorte que trata de retornar el muelle de Nitinol a su posición inicial. Esta fuerza que es constante y proporcional a medida que se extiende el muelle, afecta significativamente los tiempos de respuesta del sistema.

4.3. Experimento 3

Por tal motivo y como solución final se opta por experimentar empleando dos muelles de Nitinol (Figura 13(c)), cada muelle se encargará de deformar el otro una longitud de 11 cm. Los resultados obtenidos fueron los siguientes.

Tabla 8. Resultados de experimentar con dos muelles de Nitinol.

Calibre muelle (mm)	Tiempo 1(s)	Tiempo 2 (s)
0.75	24,42	10,96

Las columnas Tiempo 1 y Tiempo 2 indican los tiempos empleados por un muelle para deformar o estirar el muelle opuesto una longitud de 11cm, empleando o no refrigeración respectivamente. Emplear un muelle de Nitinol para reposicionar el otro ofrece una gran ventaja frente al uso de resortes.

Esto se explica porque si al energizar un muelle A, este consigue deformar totalmente al muelle B, se puede elegir dejar de energizar A sin correr el riesgo de que B trate de deformar a A, esto se debe principalmente, como se mencionó anteriormente, a que los muelles empleados tienen efecto de memoria simple, por lo que solo recuerdan una configuración definida en la fase austenita. Por lo tanto solo es posible deformar al muelle A energizando a B.

Este comportamiento brinda la posibilidad de establecer un control de posición mucho más sencillo que si se emplearan resortes, caso en el cual constantemente se debe tratar de vencer la fuerza generada por estos.

El conjunto de dos muelles se encargará de realizar la extensión y flexión de un solo dedo, por consiguiente se requieren 10 muelles de Nitinol para el sistema completo propuesto.

5. Herramientas software utilizadas

Para construir la estructura de la aplicación software encargada de la supervisión y control de las terapias, y también determinar la cantidad y tipos de herramientas a utilizar para desarrollarla, se plantearon los siguientes objetivos de diseño, teniendo en cuenta la compatibilidad entre las herramientas y los lenguajes de programación que emplean. Los objetivos son:

- Elaborar una interfaz de usuario agradable e intuitiva que permita al paciente seleccionar ejercicios y modos de trabajo para la terapia.
- Construir un modelo 3D de una mano para ser mostrado en la interfaz, de modo que el usuario cuente con una referencia visual de lo que está haciendo durante la terapia.
- Medir y registrar el desplazamiento generado por los actuadores del sistema para conocer la posición de los dedos del paciente.
- Controlar el voltaje entregado a los alambres de Nitinol teniendo en cuenta la posición de los dedos del paciente y las restricciones impuestas al sistema según su discapacidad.
- Gestionar los ejercicios que se deben hacer durante la terapia de acuerdo al paciente que se trate.

Aparte de la aplicación para las terapias, también fue necesario escoger una herramienta para el modelado y simulación de mecanismos para diseñar el robot y realizar pruebas de movilidad en las juntas del mecanismo.

Para seleccionar el software del proyecto no sólo se tuvieron en cuenta los anteriores objetivos, sino también el tipo de licencia del mismo, ya que este trabajo de investigación se encuentra enmarcado dentro del proyecto iberoamericano OpenSurg [29], cuya filosofía es desarrollar herramientas y tecnologías para robótica quirúrgica y de rehabilitación utilizando recursos software de código abierto.

5.1. Blender

Blender es un programa para el modelado en 3D destinado a artistas y profesionales de multimedia disponible bajo la licencia GNU-GPL. Como principales ventajas están la gran cantidad de herramientas para la construcción de los modelos 3D, el mapeo de texturas UV¹⁰, la posibilidad de hacer *rigging*¹¹, el generar animaciones de los modelos por medio de cinemática directa o inversa o haciendo seguimiento de curvas personalizadas, y

¹⁰ El mapeo UV consiste en usar una imagen como textura para un modelo.

¹¹ Consiste en construir elementos animables, es decir, cadenas cinemáticas que pueden ser manipuladas posteriormente, como es el caso de los esqueletos asociados a los modelos desarrollados.

renderizar¹². Adicionalmente se pueden instalar scripts para ampliar las prestaciones del programa, tal como exportadores para otros formatos. Es muy utilizado en la creación de videojuegos y al ser un programa libre y multiplataforma, la comunidad de usuarios desarrolla continuamente mejoras para el mismo y ponen a disposición de los nuevos usuarios una gran cantidad de tutoriales y ayudas para el manejo de las diferentes características del programa [30]. En la Figura 14 se muestra el modelo de una mano desarrollado en Blender.

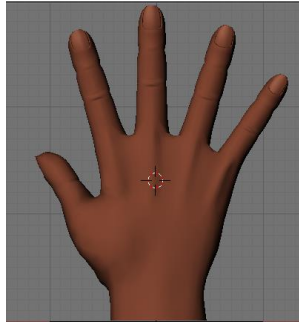


Figura 14. Modelo de una mano construida en Blender.

5.2. Autodesk Inventor

Sus principales funciones son el diseño mecánico 3D y la simulación de productos. Cuenta con herramientas para analizar la deformación plástica de moldes, para someter los modelos a pruebas de tensión y flexión con cargas específicas, realizar pruebas de movimiento de los mecanismos, generar planos de los modelos, diseñar sistemas enrutados y cableados, exportación e importación de modelos en otros formatos, y el diseño de ensamblajes a partir de piezas individuales.

La interfaz es intuitiva y muy fácil de manejar, además hay una gran cantidad de tutoriales en el sitio web de Autodesk, los cuales muestran el manejo de cada herramienta y característica de la suite.

La gran ventaja que ofrece Inventor es la posibilidad de simular los modelos construidos como si estuvieran en el entorno para el que fueron diseñados, lo cual se logra con la herramienta Sim Dynamics, en la que se crean restricciones para las juntas del mecanismo, se programan los movimientos deseados y se especifican los puntos y la magnitud de las fuerzas que se desea introducir a la simulación, así se pueden analizar los resultados de la simulación para posteriormente graficarlos y generar videos del movimiento [31]. En la Figura 15 se muestra un modelo de una mano construida en Inventor.

¹² Renderizado es el proceso de generar una imagen 3D a partir de un modelo, entendiendo modelo como una descripción en 3 dimensiones de algún objeto en un lenguaje determinado.

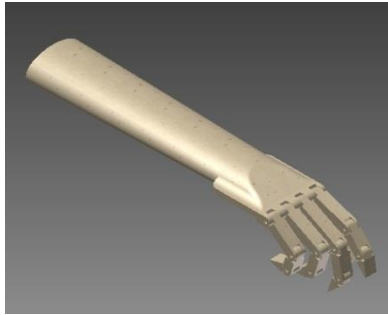


Figura 15. Ensamblaje de una mano construido en Inventor.

5.3. Ogre 3D

OGRE es la sigla en inglés de “Motor de renderizado de gráficos orientado a objetos”. Es una herramienta de código abierto, está escrito en C++ y su función principal es renderizar gráficos. Sin embargo se puede asociar fácilmente con librerías de sonido, funciones de red, dispositivos de entrada y detección de colisiones, obteniendo un poderoso motor de juegos. El único requerimiento de esta librería es que necesita de un compilador de C++, tal como Visual Studio o Code Blocks.

Ogre es un motor de renderizado en tiempo real, así las imágenes son generadas a una velocidad entre 1/30 y 1/100 fracciones de segundo por imagen, lo cual sumado a los avances en hardware de aceleración 3D, permite generar renderizados muy detallados y realistas [32].

Ogre soporta DirectX y OpenGL, y es compatible con Windows, Linux y Mac OSX. Permite cargar materiales y texturas diseñadas en otros programas, aplicar edición a los sombreados, mezclar texturas y hacer operaciones con color. Permite importar mallas desde muchos formatos y realizar animaciones esqueléticas desde cero o juntando las animaciones importadas con la malla. Usa clases predefinidas para la organización de la escena, tiene plugins dedicados al renderizado de terrenos e interiores y permite enlazar objetos y seguir sus movimientos por medio de nodos de escena [33]. En la Figura 16 se muestra un modelo de una mano renderizada con Ogre.



Figura 16. Mano renderizada en Ogre.

5.4. Qt

Es una biblioteca multiplataforma utilizada principalmente para el desarrollo de aplicaciones con una interfaz gráfica de usuario. Es producido por la división de software de Nokia pero es software libre, así se desarrolla más rápido que otras herramientas libres porque además de contar con los desarrollos de la comunidad, también es financiado por una empresa privada. Para crear aplicaciones con Qt basta con tener conocimientos en el lenguaje de programación C++. Su entorno de desarrollo se llama Qt Creator, y tiene la ventaja que permite editar visualmente las interfaces diseñadas, lo cual se ve reflejado en lo llamativas y originales que resultan ya que facilita el ajuste de las dimensiones de botones y ventanas y su ubicación en la interfaz final [34].

Algunas aplicaciones que utilizan Qt son el reproductor VLC, el entorno de escritorio KDE, Google Earth y Skype, entre otros, además de encontrarse en la mayoría de smartphones Symbian. Además, cuenta con gran cantidad de *bindings*¹³ para diversos lenguajes de programación tal como Java, Python, Pascal, Ada, o PHP, por mencionar algunos [35].

En la Figura 17 se muestra un diseño previo de la interfaz de usuario para el presente proyecto desarrollado con Qt.



Figura 17. Interfaz de usuario diseñada en Qt.

5.5. Estructura de la aplicación

Los objetivos de diseño de la aplicación software para el presente proyecto se pueden cumplir por medio de las herramientas descritas en los párrafos anteriores. En Ogre se lleva a cabo la lectura de la información proveniente de los sensores y el control del paso de energía a los actuadores, lo cual se logró integrando Ogre con una librería para comunicación serial que permite el intercambio de datos entre la aplicación y la tarjeta de acondicionamiento y control del sistema.

Blender se utiliza únicamente para construir el modelo 3D de la mano ya que su interfaz gráfica para edición de mallas permite apreciar las modificaciones realizadas al modelo mientras se llevan a cabo, contrario a lo que sucede con Ogre, en donde los modelos se construyen con funciones y sólo se pueden apreciar los cambios y resultados hasta

¹³ *Binding* es una adaptación de una biblioteca que permite usarla en un lenguaje de programación diferente del lenguaje en que está escrita.

compilar y ejecutar la aplicación. Otra ventaja de Blender es que los modelos tienen mejor aspecto y mayor nivel de detalle que aquellos elaborados en Ogre. Así, simplemente se requiere transformar el modelo desarrollado con Blender a un formato que entienda Ogre, lo cual se realiza con un exportador de mallas.

Para desarrollar la interfaz de usuario, aunque las ventanas y botones que se pueden diseñar con Ogre no tienen mala presentación, en realidad son algo rústicos y sencillos, lo cual es suficiente para una aplicación orientada a ingenieros, en la que se desea leer valores y conocer el estado de una planta sin importar el aspecto de la interfaz, pero tal vez sean poco llamativos y amigables para una aplicación cuyos usuarios necesitan una experiencia tan agradable como sea posible. Por esta razón se decidió utilizar Qt para construir la interfaz de usuario, ya que permite lograr componentes gráficos con mejor presentación y diseño que los construidos con Ogre.

Para simular el mecanismo se decidió utilizar Autodesk Inventor, ya que si bien es un software propietario, existe una versión estudiantil cuya licencia es gratuita y no limita las prestaciones del programa.

6. Diseño e implementación del sistema

6.1. Sistema de rehabilitación propuesto

El objetivo que se persigue con las terapias de rehabilitación es que los pacientes que han sufrido accidentes cerebro-vasculares recuperen el control del movimiento de sus miembros afectados, lo cual es algo más difícil en el caso de las manos porque requiere más trabajo el recuperar la precisión y funcionalidad que las caracteriza.

Como se mencionó en la sección 2.2, tras el ACV muchos de los pacientes presentan hipertonia muscular, es decir que sus músculos afectados se mantienen tensionados todo el tiempo, razón por la cual los pacientes pueden experimentar dolor al tratar de extender sus músculos durante las terapias físicas, y también hay que recordar que es más rápida la recuperación en la fuerza de los flexores que la de los extensores. A causa de la hipertonia muscular, un dispositivo que asista las terapias para la apertura de la mano debe ser capaz no sólo de superar la fuerza en contra que ejercen los flexores de la mano, sino también considerar que, inicialmente, cada dedo permite un rango de movimiento diferente, o sea que unos dedos ofrecerán mayor resistencia que otros al ser extendidos.

Estos factores condicionan el diseño y principio de funcionamiento de un dispositivo asistente para terapias de rehabilitación. Por esta razón en la presente investigación se plantea la siguiente solución como dispositivo asistente para terapias de rehabilitación en pacientes con dificultades de movilidad en los dedos de sus manos.

El sistema debe servir como soporte para el brazo del paciente y permitir también el bloqueo de la muñeca, de modo que el único movimiento posible se dé en los dedos. Para realizar la extensión de los dedos se propone tomarlos por el extremo distal y empujarlos desde debajo de la mano, con el fin de extenderlos en la medida que el paciente soporte (Figura 18). Otra consideración es que los dedos se encuentran flexionados inicialmente,

por lo cual el dispositivo debe ser capaz de fijarse a los dedos estando éstos algo cerrados y aplicar desde ahí la fuerza para extenderlos en la dirección deseada.

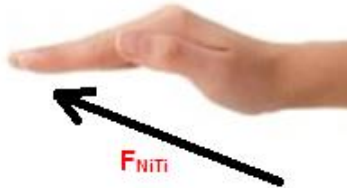


Figura 18. Dirección de aplicación de la fuerza del Nitinol.

Como actuadores del sistema se utilizan muelles de Nitinol, los cuales presentan características de fuerza y desplazamiento mayores en comparación con los alambres tradicionales. Sin embargo hay que tener en cuenta que la fuerza generada por dichos muelles es lineal, así es necesario encontrar la dirección apropiada para aplicarla y lograr la extensión, la cual se muestra en la Figura 18, de tal forma que se extiendan primero las articulaciones metacarpianas y posteriormente las interfalángicas, así el plano de movimiento del Nitinol estaría inclinado respecto al plano de la mano. Este ángulo de inclinación se determinó por medio de simulación con la herramienta Sim Dynamics de Autodesk Inventor, en donde se comparó el comportamiento de los diferentes ángulos mientras se extendían los dedos.

Para supervisar el desempeño del sistema, se plantea enlazar el dispositivo mecánico a una aplicación software que controle el paso de energía a los actuadores y que indique al paciente cómo realizar la terapia, haciéndolo sentir cómodo con lo que se muestra en la interfaz, y que le aporte datos referentes a la terapia, tal como el tiempo transcurrido o las repeticiones que le falta realizar.

En la Figura 19 se muestra el modelo 3D del mecanismo y la mano del paciente. Para las medidas de la mano se tomaron como referencia las de la mano UC-1 de la Universidad del Cauca [36].

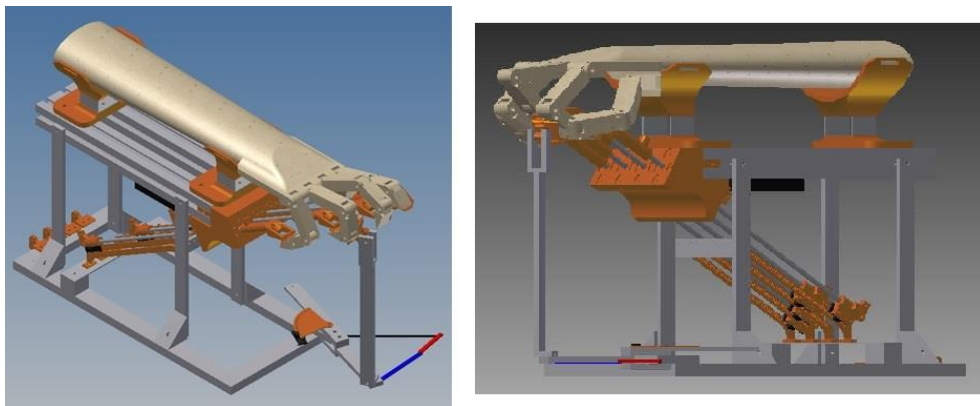


Figura 19. Ensamblaje del robot y la mano en Inventor.

Como se puede apreciar en la Figura 19, es necesario utilizar un medio para transmitir la fuerza generada por el Nitinol a los dedos, para lo cual se utilizan varillas de aluminio, así uno de sus extremos se fija al dedo y el otro al Nitinol.

En este caso, como en otros dispositivos para rehabilitación de manos, también es necesario que el paciente permita la apertura de su mano, lo cual puede ser realizado por el fisioterapeuta o por el mismo paciente, ya que se requiere que los dedos estén parcialmente extendidos para poder colocar los sujetadores para la punta de los dedos y así utilizar el dispositivo.

El diseño propuesto como solución se describe en detalle a continuación, en la subsección 6.2, “Diseño CAD del sistema”.

6.2. Diseño CAD del sistema

La simulación y el diseño de las piezas del mecanismo se realizaron con el software CAD Autodesk Inventor Professional 2012, versión estudiantil. Algunas piezas del mecanismo cuya forma era difícil de moldear, fueron elaboradas en la máquina de prototipado rápido 3D Systems ProJet HD 3000 [37]. Estas piezas se exportaron desde Inventor en formato stl, el cual conserva del objeto 3D sólo la información sobre su superficie geométrica, haciendo de lado características como el color o la textura presentes en el modelado.



Figura 20. Impresora ProJet HD 3000.
Fuente: [37]

Como ya se mencionó, en el presente proyecto se emplean muelles de Nitinol como actuadores del sistema. Así el primer requerimiento de diseño es inclinarlos respecto al plano de la mano para lograr la extensión de los dedos del paciente ya que generan un desplazamiento lineal. Como restricción surgió el hecho que para el cambio de fase de estos actuadores es necesario alcanzar temperaturas tales que pueden generar quemaduras al paciente, así que es mejor aislarlos para evitar posibles lesiones. Además, hacía falta transmitir el desplazamiento generado por el sistema de actuación a cada uno de los dedos, para lo cual se pensó en una pieza por cada dedo que se deslizara a través de un canal con baja fricción en la tapa del mecanismo, la cual cubriría los alambres. Para encontrar el ángulo de inclinación se construyó un primer modelo en el que se realizaron una serie de simulaciones, en cada una de las cuales se varió el valor del ángulo de inclinación de la pieza que se desliza. Este modelo consistía sólo del dedo índice y se muestra en la Figura 21. En ella se puede apreciar la pieza que se desliza en color negro, la base del mecanismo y soporte de la mano en color azul, y el dedo índice en color piel.

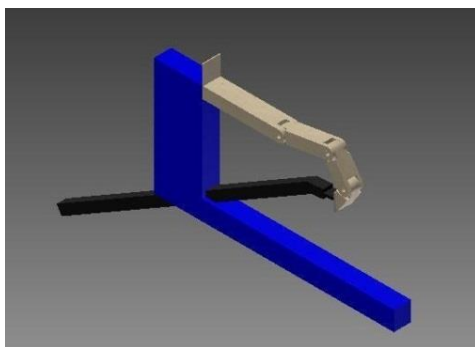


Figura 21. Modelo de pruebas para el ángulo de inclinación de los actuadores.

El criterio utilizado para determinar el ángulo de inclinación óptimo, consistió en vigilar que las articulaciones del dedo no giraran hasta posiciones fuera del rango de movimiento normal para un adulto promedio, el cual se introdujo en la Tabla 1, al inicio del presente documento, de la cual se muestra la información relevante para las simulaciones a continuación en la Tabla 9.

Tabla 9. Rango de movimiento articular para el sistema.

Dedos	Articulación	Tipo de movimiento	Rango de movimiento
Meñique al índice	DIP	flexión-extensión	0° a 85°
	PIP	flexión-extensión	-10° a 120°
	MCP	flexión-extensión	-80° a 120°
Pulgar	IP	flexión-extensión	-45° a 90°
	MCP	flexión-extensión	-45° a 90°
	CMC	aducción-abducción	-10° a 45°

Para vigilar la variación de estos ángulos durante la simulación se utilizó la herramienta “Gráfico de salida” ofrecida en Sym Dynamics de Autodesk Inventor. Las simulaciones se realizaron para una inclinación entre 20° y 50°, dando como resultado un óptimo de 30°, ya que para esta inclinación, se logró la extensión completa del dedo, se respetó el rango de movimiento de cada articulación y el desplazamiento lineal necesario para realizar esta extensión fue de 11 cm, lo cual no es un valor muy alto en comparación con otras simulaciones.

Los resultados de la simulación con inclinación 30° se muestran a continuación en la Figura 22, Figura 23 y Figura 24, aclarando que cuando las articulaciones están completamente extendidas su posición es 0°, y a medida que se flexionan, su posición va aumentando.

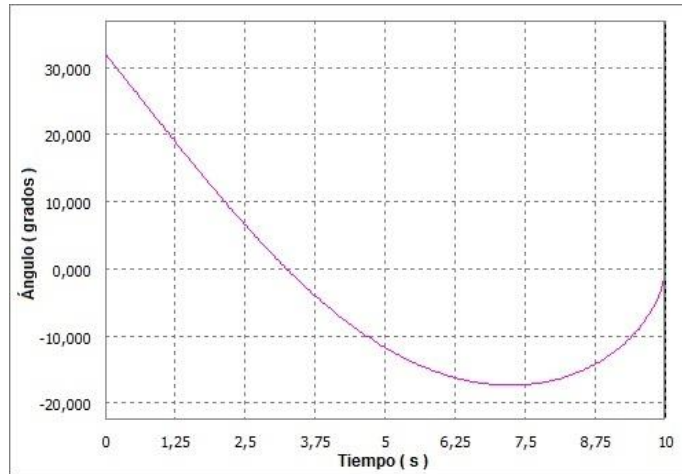


Figura 22. Resultados para la articulación metacarpofalángica del dedo índice.

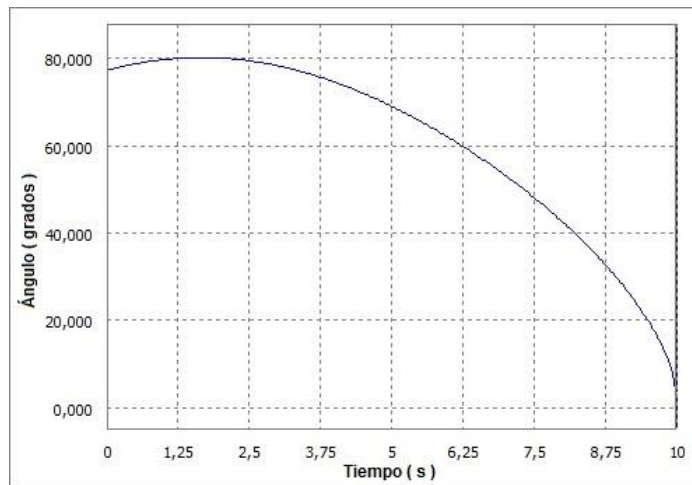


Figura 23. Resultados para la articulación interfalángica proximal del dedo índice.

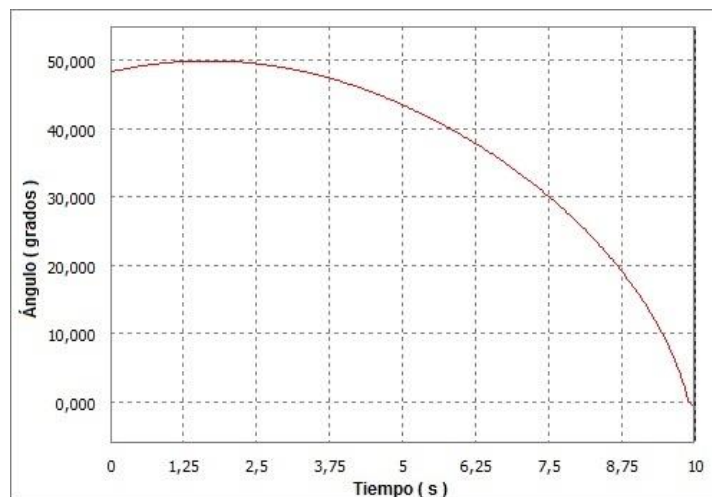


Figura 24. Resultados para la articulación interfalángica distal del dedo índice.

En estos resultados de simulación se puede observar que la posición de inicio de cada articulación es diferente, siendo la articulación más flexionada la interfalángica proximal con 80° . Además, en el caso de la articulación metacarpofalángica, se logra una hiperextensión de 15° , lo cual permite cubrir una mayor parte del rango de movimiento para esta articulación, el cual está entre -80° y 120° según la Tabla 9. También se puede apreciar que primero se extiende casi por completo la articulación metacarpofalángica antes de realizar extensiones notables en las articulaciones interfalángicas, con lo cual se cumplió el criterio de diseño planteado en la sección 6.1, "Sistema de rehabilitación propuesto".

Una vez establecido el ángulo en que debían estar inclinados los actuadores se procedió a modelar las piezas de todo el mecanismo, tanto aquellas que iban a ser impresas como aquellas que se iban a implementar con otros materiales.

Lo primero que se modeló fue la mano del paciente para construir de acuerdo a sus dimensiones y forma, las piezas que se ajustan a los dedos (Figura 25 (a)). A cada una de estas piezas se le agregó una protrusión por la que pasa un eje que permite la rotación de las piezas a la par con las falanges distales de los dedos, ya que están fijas a éstas, y mantienen a la vez la conexión con el sistema de actuación sin esfuerzo adicional para los actuadores. Esta protrusión se muestra en color azul en la Figura 25 (b).

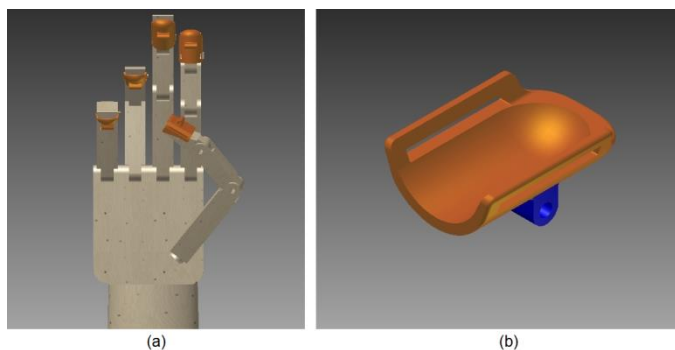


Figura 25. Sujetadores de los dedos.

Para transmitir la fuerza generada por el Nitinol a los dedos se decidió utilizar varillas de aluminio, así se modelaron también unas piezas que sirven como puente entre las varillas y las piezas que sujetan los dedos, y su forma se ve condicionada por el espacio que hay entre los dedos de la mano y la protrusión mencionada en el párrafo anterior (Figura 26).

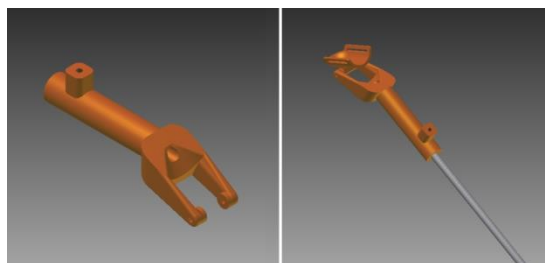


Figura 26. Sistema de transmisión de fuerza para los dedos 2 al 5.

Al igual que sucedió con el ángulo de inclinación de las varillas, por medio de simulación se determinó la altura a la que debía ubicarse la mano del paciente respecto de la parte superior del sistema, ya que a mayor altura, debía aumentarse el desplazamiento de las varillas para lograr un rango de movimiento aceptable. Así se modeló un par de piezas para soportar la muñeca y el brazo del paciente, con las que además se consigue bloquear el movimiento de la articulación de la muñeca (Figura 27 (a)). Además se diseñaron un par de piezas que sirven como base para los soportes del brazo, cuya finalidad es que permitan cambiar la posición de estos puntos de apoyo para que el sistema se ajuste a diferentes tamaños de brazo (Figura 27 (b)). En vista que estas piezas se ubican sobre la parte superior del mecanismo, fue necesario encontrar una tapa con rieles para lograr el desplazamiento de las piezas de soporte pero a la vez evitar desprenderlas del sistema (Figura 27 (c)).

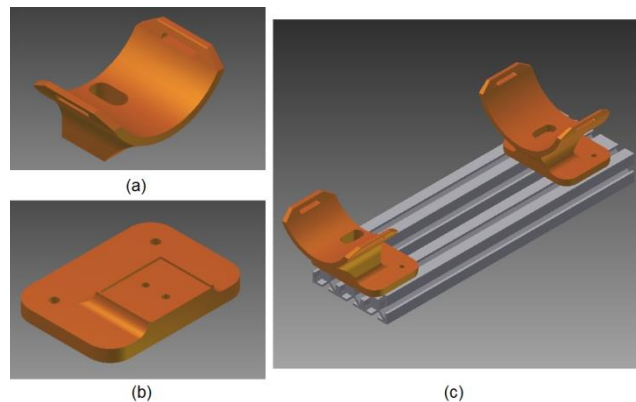


Figura 27. Soportes para la muñeca y el antebrazo.

Para lograr baja fricción en el deslizamiento de las varillas, se utilizaron los cojinetes Igubal EFOM (Figura 28), cuyo coeficiente de fricción dinámico con el metal varía de 0.08 a 0.23, son muy resistentes al desgaste y no requieren limpieza para conservar sus propiedades [38]. Sin embargo, estos componentes son poco profundos como para usar sólo uno como canal para las varillas, así se optó por emplear 2 cojinetes por cada varilla, con el fin de tener estabilidad. Para esto se diseñó una pieza que permitiera montar cojinetes en 2 caras opuestas (Figura 29 (a)), y se atornillaron los componentes a la pieza diseñada aprovechando que la carcasa de los cojinetes cuenta con 2 orificios para fijación. Además se adicionaron 4 protrusiones para ajustar esta pieza a la tapa escogida para el dispositivo, la cual cuenta con rieles en todas sus caras (Figura 29 (b)).

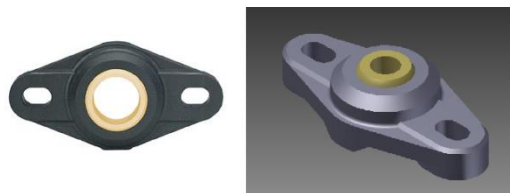


Figura 28. Cojinete Igubal EFOM - Modelo real y modelo CAD.

Fuente: [38]

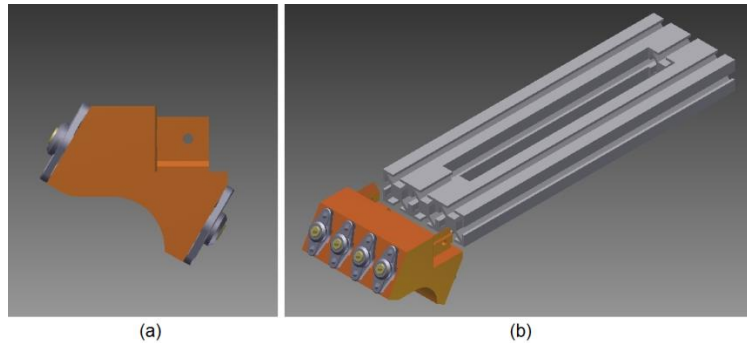


Figura 29. Base para los cojinetes.

En la sección de experimentos se concluyó que para poder acelerar la respuesta del sistema realizando las tareas de extensión y flexión se emplearán dos muelles de Nitinol por dedo que realicen la extensión y flexión del mismo. Ahora bien, la fuerza generada por el Nitinol se aplica a las varillas en un extremo de las mismas, por lo cual se diseñó una pieza que se monta en este punto de la varilla y sirve como punto de conexión para los 2 alambres que actúan en cada dedo. También se diseñaron otros 2 modelos de piezas para fijar los extremos libres de los alambres a la estructura. Todas estas piezas se pueden apreciar en la Figura 30, en la cual los alambres de Nitinol se muestran en color verde.

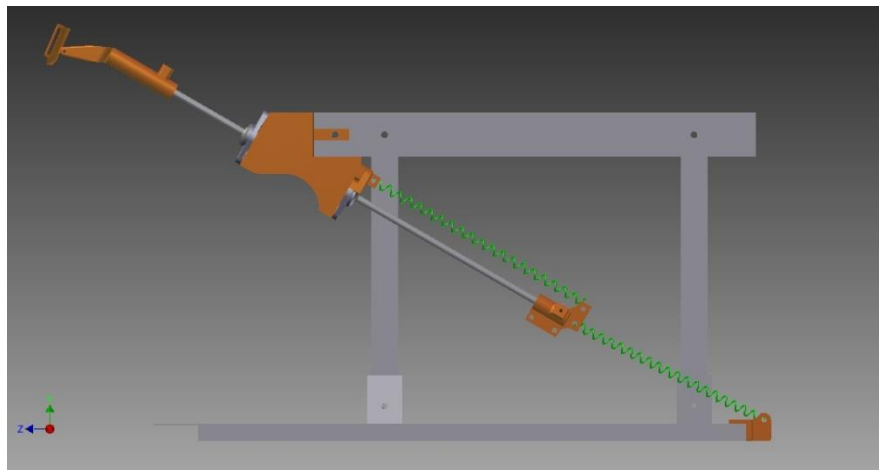


Figura 30. Fijación del Nitinol al sistema.

Sin embargo, es necesario mencionar que los muelles de Nitinol no fueron simulados como actuadores en Sim Dynamics, ya que su constante de elasticidad varía con el cambio de fase, y el asistente para el modelado de resortes de Inventor no permite modelar esta variabilidad. Lo más cercano para simular el Nitinol es lo que se muestra en la Figura 30, que consiste en una espiral extruida, cuya deformación se puede animar pero que no se puede simular a la par con el ensamblaje del resto de piezas, razón por la cual en Sim Dynamics no se incluyeron los alambres de Nitinol sino que se programaron los movimientos a realizar en las juntas generadas entre las varillas y la pieza que sirve como base para los cojinetes de baja fricción.

Para diseñar el sistema que mueve el dedo pulgar fue necesario considerar que este dedo se mueve en un plano distinto al de los otros 4 dedos. Entonces no resultaba viable utilizar una varilla para transmitir la fuerza del Nitinol al dedo porque no era posible ubicarla sin entrar en conflicto con el desplazamiento de las otras varillas. Sin embargo, al observar el proceso de extensión y flexión del pulgar se pudo apreciar que el extremo distal del mismo describe un arco, cuyo radio se puede determinar con la ayuda de las dimensiones de las falanges del dedo (Figura 31).

Ahora bien, ya que se deben usar muelles de Nitinol como actuadores del sistema, el modo más práctico para generar el arco a partir de un movimiento lineal es haciendo rotar una barra sobre un punto de pivote, cuya posición depende del centro del arco deseado, el cual se ubicó inicialmente en la articulación metacarpofalángica del pulgar y se ajustó posteriormente mediante simulación (Figura 32).

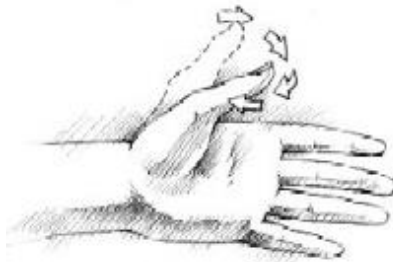


Figura 31. Flexo-extensión del pulgar.
Fuente: [39]

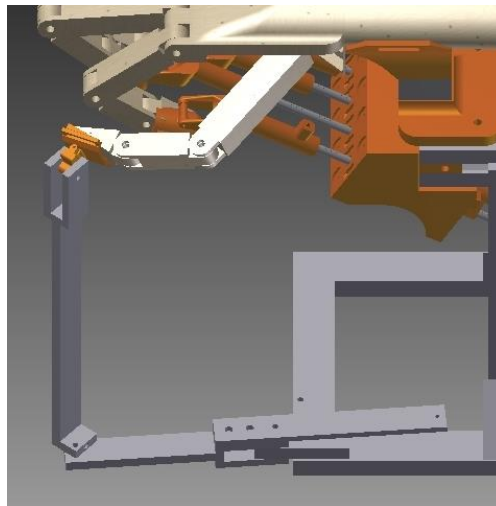


Figura 32. Barra para el movimiento del pulgar.

Cabe mencionar que, una mano espástica normalmente luce como la Figura 33, en donde se puede apreciar que los dedos índice al meñique aprisionan el dedo pulgar. Por esta razón, en el caso de la articulación carpometacarpiana del dedo pulgar, no se rehabilita su flexión y extensión, sino su aducción y abducción, ya que así se aleja este dedo del eje central de la mano, liberándolo de los otros 4 dedos, además de crear conciencia en el cerebro del paciente de la forma que debe tomar la mano al abrirla en su totalidad.

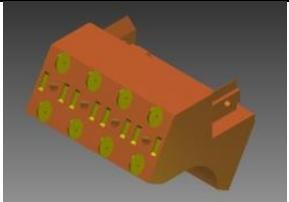
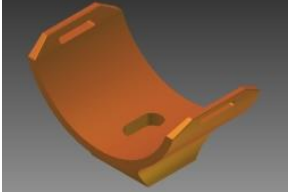


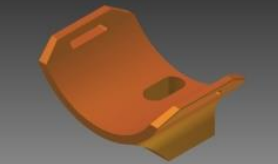
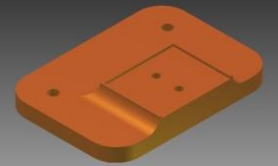
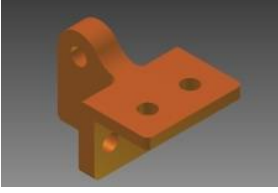
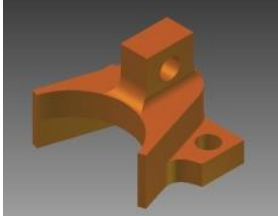
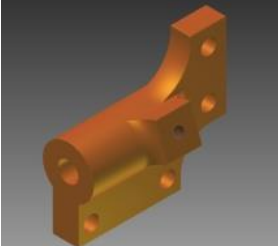
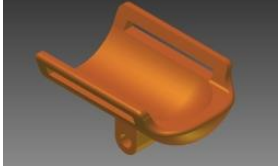

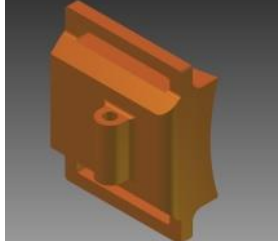
Figura 33. Mano espástica.
Fuente: [39]

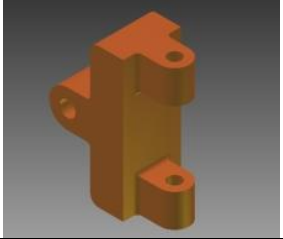
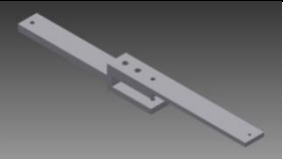
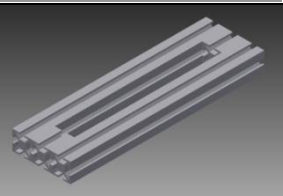
Así se completaron todas las piezas del mecanismo, procediendo con la simulación. Con las pruebas en Sim Dynamics se evaluó el movimiento del mecanismo, para lo cual se observaron las posiciones que tomaban los dedos de la mano modelada en cada instante de la simulación. Una vez verificado que no se violaban los rangos de movimiento expresados en la Tabla 9, se procedió a imprimir las piezas en la impresora 3D Systems Projet HD 3000. Esta impresora utiliza 2 tipos de material para las impresiones. Uno es el Visijet SR200 [40], componente principal de las piezas impresas, por lo cual se evaluaron sus propiedades físicas para determinar si era necesario reforzar posteriormente las piezas diseñadas para que soportaran las condiciones de trabajo deseadas. En este proyecto, la propiedad física de mayor importancia es el punto de fusión/congelamiento de las piezas, el cual se da entre 55°C y 65°C, y en vista de que la temperatura de transformación de los muelles de Nitinol se encuentra entre 44 y 55°C, es necesario aislar las piezas de modo que el calor no las deforme. El otro tipo de material es el Visijet S100 [40], que consiste en cera que sirve como material de soporte para la impresión, así todos los espacios vacíos, bien sea propios de las piezas o el espacio entre ellas se rellena con esta cera. Al finalizar la impresión este material se remueve y desecha.

En la Tabla 10 se muestra un resumen de las piezas diseñadas para el mecanismo y un código asociado a cada una, con el fin que sea fácil referirse a las piezas en secciones posteriores, y en la Figura 34 se muestra el sistema completo que se implementó en Inventor.

Tabla 10. Piezas diseñadas para el mecanismo.

Referencia	Imagen	Descripción
BC (Base cojinetes)		Pieza para fijar los cojinetes de baja fricción.
SA (Soporte antebrazo)		Soporte para apoyar el antebrazo del paciente.

<p>SM (Soporte muñeca)</p>		<p>Soporte para apoyar y bloquear la muñeca del paciente.</p>
<p>BSM,BSA (Bases de soporte muñeca y antebrazo)</p>		<p>Bases para cambiar los puntos de apoyo del antebrazo y la muñeca respectivamente.</p>
<p>BIN2-BIN5 (Base inferior Nitinol 2 a 5)</p>		<p>Piezas para sujetar el extremo inferior de los cables de Nitinol para flexión a la base del mecanismo. El número corresponde al dedo al que están asociadas.</p>
<p>BSN2-BSN5 (Base superior Nitinol 2 a 5)</p>		<p>Piezas para sujetar el extremo superior de los cables de Nitinol para extensión a la pieza BC. El número corresponde al dedo al que están asociadas.</p>
<p>BUN2-BUN5 (Base de unión Nitinol 2 a 5)</p>		<p>Piezas de unión del extremo inferior de las varillas y el extremo superior e inferior de los cables de Nitinol para flexión y extensión respectivamente. El número corresponde al dedo al que están asociadas.</p>
<p>AD2-AD5 (Apoyo dedos 2 a 5)</p>		<p>Piezas para fijar los dedos al mecanismo. El número corresponde al dedo al que están asociadas.</p>
<p>SAD2-SAD5 (Soporte apoyo dedos 2 a 5)</p>		<p>Piezas de unión del extremo superior de las varillas. El número corresponde al dedo al que están asociados.</p>
<p>AD1 (Apoyo dedo 1)</p>		<p>Pieza para fijar el dedo pulgar al mecanismo. El número uno corresponde al dedo asociado al que la pieza está asociada, que es el pulgar.</p>

<p>CAD1 (Complemento AD1)</p>		<p>Pieza que permite el giro sobre el eje longitudinal de la mano de la pieza AD1.</p>
<p>BD1 (Base dedo 1)</p>		<p>Pieza de aluminio para transmitir la fuerza de los actuadores del pulgar y generar la aducción-abducción de este dedo.</p>
<p>SSA (Soporte superior de aluminio)</p>		<p>Soporte de aluminio que permite sujetar la pieza BJ y permite deslizar los soportes para la muñeca (SM) y el antebrazo (SA).</p>

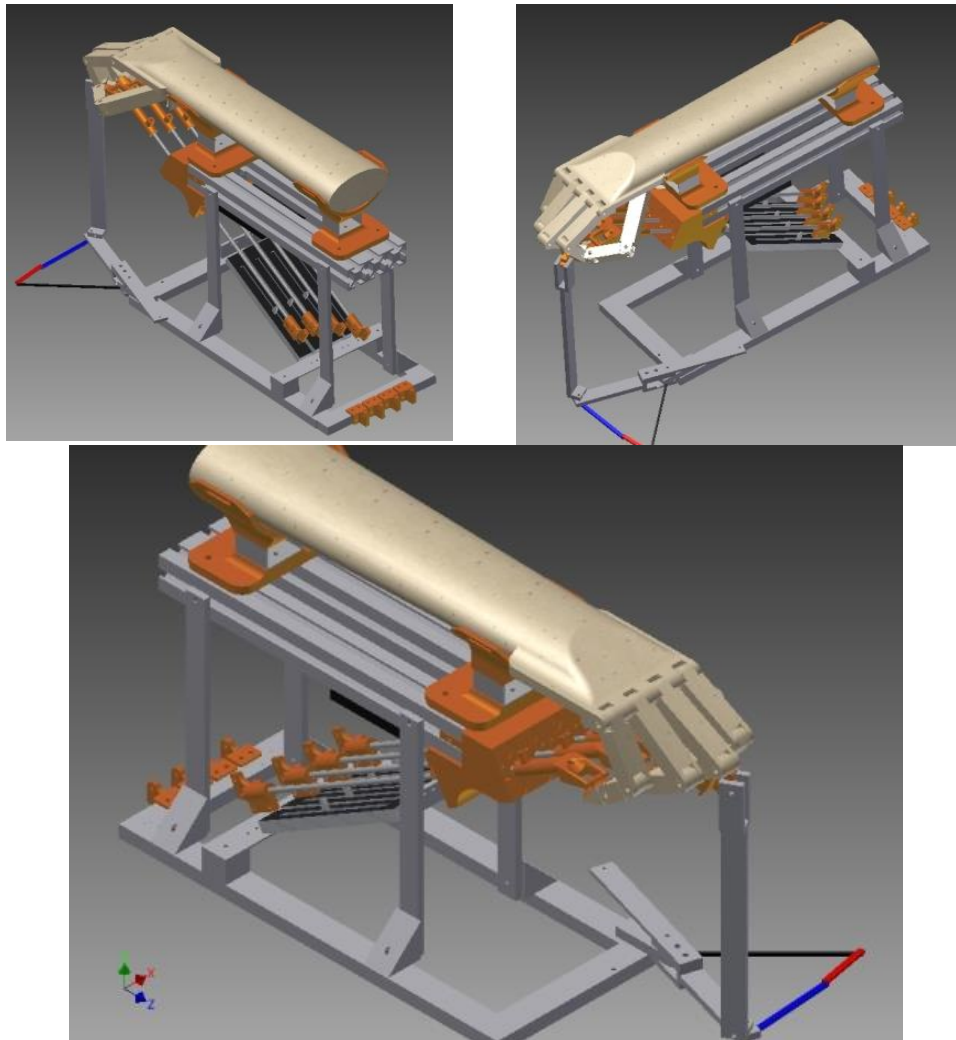


Figura 34. Sistema completo implementado en Inventor.

6.3. Implementación del sistema mecánico

Todos los componentes mecánicos a excepción de las piezas elaboradas por la impresora 3D son de aluminio, un material resistente, liviano y anticorrosivo, capaz de proporcionar un soporte estructural sólido para las piezas fabricadas por la impresora, además de soportar el peso que ejercerá el paciente sobre el sistema.

La descripción del proceso realizado para el ensamble del sistema mecánico se presenta en el ANEXO C.

A continuación se muestran algunas imágenes del sistema fabricado:

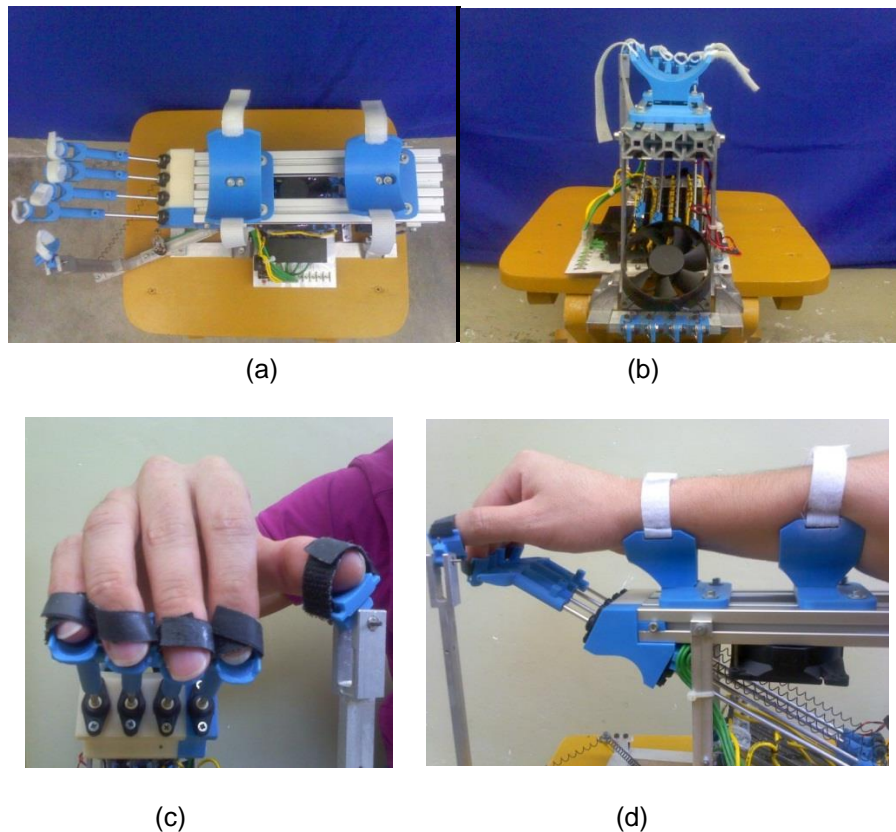


Figura 35. Vistas del sistema fabricado.

6.4. Tarjeta de control y sensores

6.4.1. Tarjeta de control

Se opta por adquirir dentro de la amplia gama de tarjetas marca Arduino el modelo Arduino UNO [41]. La ventaja de las tarjetas Arduino frente a otras marcas radica principalmente en tres factores: su bajo costo, sencillez de programación, y porque es

una plataforma electrónica abierta lo que permite encontrar miles de personas alrededor del mundo que desarrollan y comparten libremente aplicaciones bajo esta plataforma.

La elección no fue difícil ya que todas las tarjetas Arduino están construidas bajo versiones diferentes de un mismo micro-controlador, el ATmega que supera en su versión más básica los requerimientos de hardware necesarios para controlar el sistema de rehabilitación propuesto. Las diferentes versiones de este micro-controlador permiten hacer variaciones en el número de salidas digitales y capacidad de almacenamiento. Los requerimientos mínimos de hardware establecidos para controlar el sistema se presentan a continuación:

Tabla 11. Especificaciones para controlar el sistema propuesto

Requerimientos	Cantidad	Comentario
Salidas PWM	5	Empleadas en la fase de extensión
Entradas analógicas	5	Empleadas para los sensores
Salidas digitales	7	5 empleadas en la fase de flexión y 2 para refrigeración.
Programación sin accesorios adicionales	NA	Programación rápida
Rápido montaje	NA	Fácil cableado para realizar experimentos
Fácil reparación	NA	Reemplazar componentes averiados fácilmente

Por lo tanto, la elección fue determinada por la simplicidad y facilidad con que se pudieran montar y desmontar componentes para realizar fácilmente los experimentos con actuadores de Nitinol. Así se opta por seleccionar la tarjeta Arduino UNO. Sus especificaciones técnicas se presentan a continuación:

Tabla 12. Especificaciones tarjeta Arduino UNO
Fuente [41].

Especificaciones	Valor
Micro-controlador	ATmega328
Salidas digitales	14
Salidas PWM ¹⁴	6
Entradas analógicas	6
Velocidad reloj	16 MHz
Corriente DC por pin	40 mA

Como se aprecia en la tabla anterior, se pueden configurar 5 salidas PWM para poder energizar los alambres de Nitinol durante la fase extensión, y cuenta con las suficientes salidas digitales para energizar los alambres de Nitinol en la fase de flexión, así como para energizar los ventiladores. Además cuenta con 6 entradas analógicas de las cuales

¹⁴ Las salidas PWM se obtienen de configurar las salidas digitales que ofrecen las tarjetas Arduino, por consiguiente se deben restar las salidas PWM que se emplearán del total de salidas digitales, para obtener las salidas de este tipo que se tendrían libres.

se emplean 5 para la lectura de los sensores empleados para determinar la ubicación de los dedos.



Figura 36. Tarjeta Arduino Seleccionada
Fuente: [41]

En la página oficial de Arduino [41] se encuentra la información necesaria para programar la tarjeta. Además se encuentra un paquete que permite comandar la tarjeta Arduino desde Matlab, esto abre las posibilidades para crear algoritmos de control mucho más elaborados y con alta carga computacional que no podrían ejecutarse directamente desde la tarjeta así como muchas otras aplicaciones.

6.4.2. Sensores

Para elegir el dispositivo de sensado que permitiera determinar la posición de cada dedo durante todo el recorrido, se tuvieron en cuenta tres factores:

- Simplicidad: se deseaba un elemento que no exigiera ningún tipo de acondicionamiento mecánico para su funcionamiento.
- Peso reducido: con el fin de no incorporar peso adicional al sistema, en caso de que el sensor necesitara ser desplazado a medida que se extienden o flexionan los dedos.
- Acondicionamiento de señal simple: se desea que el sistema de sensado no requiera electrónica adicional para acondicionar la señal entregada.

Establecidos estos criterios se plantearon dos alternativas. En la primera, se pretendió hacer uso de una regleta perforada ubicada por debajo de cada tubo de aluminio (Figura 38). Las piezas BUN2 a BUN5 tendrían sujeto un encoder incremental de dos canales (Figura 37) que permitirían contar cada ranura de la regleta perforada a medida que se desplazan, y así determinar la posición de cada dedo, como se muestra en el diseño preliminar construido en Autodesk Inventor.



Figura 37. Encoder incremental de dos canales.
Fuente: [42]

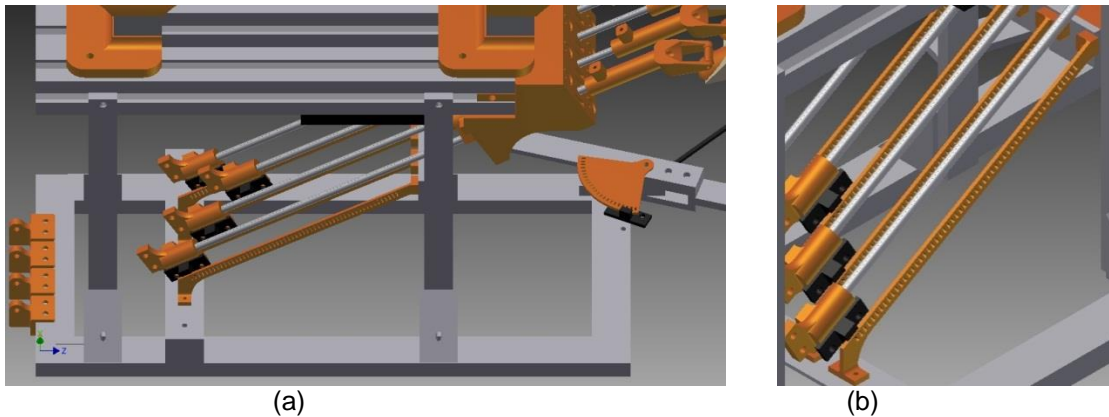


Figura 38. Vistas 3D del sistema con una regleta perforada y encoder incremental.

El sistema a pesar de ser sencillo y liviano, presentó un inconveniente. Los encoders a emplear brindan la posibilidad de que el usuario conozca en qué sentido se desplaza el sensor, para ello disponen de dos canales o salidas digitales que entregan una secuencia binaria diferente según el sentido de movimiento del sensor [42]. Es necesario disponer de dos entradas digitales que permitan procesar esta codificación y así determinar en qué sentido se está desplazando. En total se requieren de 10 entradas digitales disponibles en la tarjeta Arduino UNO, con las que no se cuentan.

La segunda alternativa, consiste en emplear potenciómetros de deslizamiento (Figura 39) que permitan un recorrido mínimo de los 11 cm necesarios para poder controlar la posición de cada dedo.



Figura 39. Potenciómetro deslizante de 100K
Fuente: [43].

Esta alternativa es simple, de fácil acondicionamiento, permite una fácil integración con el sistema propuesto, ya que solo basta ubicarlos debajo de cada varilla y sujetar la pieza deslizante del potenciómetro a la pieza BUNx¹⁵ y solo se requiere de una entrada analógica por dedo para determinar su posición. Por estos motivos fue elegida como sistema de sensado.

¹⁵ x representa el número de pieza para los dedos 2 a 4.

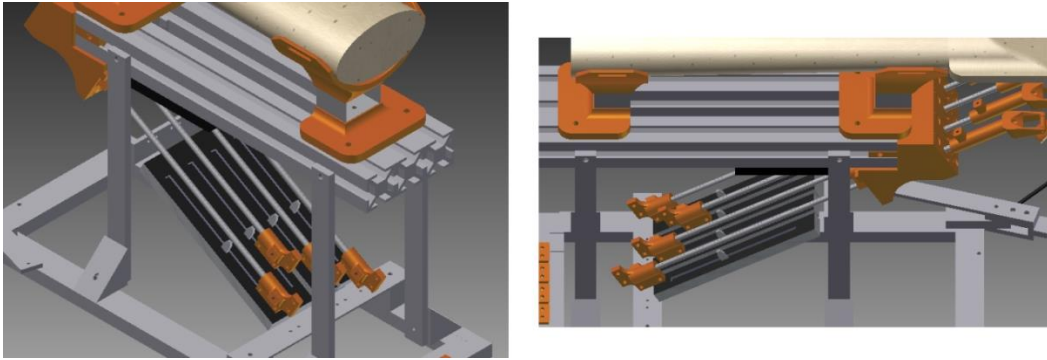


Figura 40. Potenciómetros ensamblados en Inventor.

6.5. Diseño y construcción de la tarjeta de acondicionamiento de señales

6.5.1. Acondicionamiento de señales

Se llama acondicionamiento de señales al proceso de manipular las señales con el propósito de ajustarlas a los niveles y características óptimos que requiere la siguiente etapa de un proceso a realizar.

En este trabajo fue necesario el acondicionamiento de dos señales. Inicialmente la señal entregada por los potenciómetros de deslizamiento, con el propósito de que entregaran una variación de voltaje en un rango de 0-5 Voltios DC apropiada para las entradas analógicas de la tarjeta Arduino UNO. Para lograrlo se empleó un divisor de tensión (Figura 41).

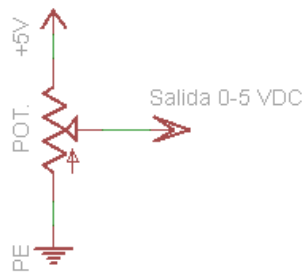


Figura 41. Esquemático de un divisor de tensión 0-5 VDC

Además fue necesario acondicionar las salidas PWM y digitales de la tarjeta Arduino, con el fin de suministrar la potencia requerida. Para ello se emplearon transistores 2N3904, 2N3906 y Mosfet IRF530 cuyas especificaciones técnicas (ANEXO A) cumplen las demandas de corriente exigidas para energizar los muelles de Nitinol y los ventiladores.

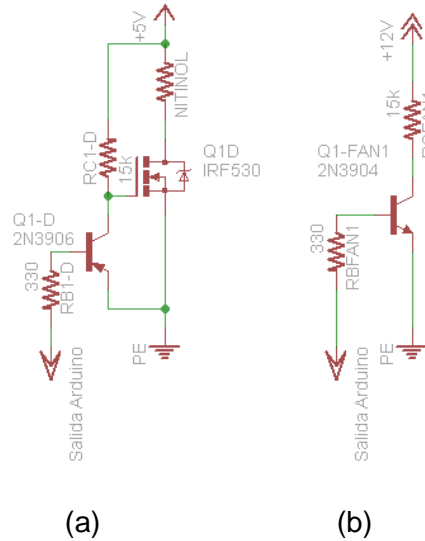


Figura 42. Esquemáticos de acondicionamiento.

El transistor 2N3906 y Mosfet IRF530 se emplean para energizar los muelles de Nitinol (Figura 42 (a)). Esta configuración permite brindar protección de sobrecarga a las salidas de la tarjeta Arduino UNO capaz de proporcionar solo 40 mA por pin.

En esta configuración es el transistor 2N3904 el que soporta la corriente necesaria para activar el mosfet de potencia, el transistor es capaz de soportar 200mA de corriente a través de su colector.

Para energizar los ventiladores a 12 Vdc y 150 mA se emplearon transistores 2N3904 como interruptores (Figura 42 (b)), estos al igual que el 3904 son capaces de soportar corrientes de hasta 200mA.

Estas configuraciones fueron probadas en una portoboard (tarjeta de prototipado), para verificar su funcionamiento antes de su implementación (Figura 43).



Figura 43. Verificación experimental de las etapas de acondicionamiento.

6.5.2. Diseño de la tarjeta de acondicionamiento

Se opta por diseñar una tarjeta de circuito impreso en la que se implementa un número determinado de veces las etapas de acondicionamiento, es decir la etapa de acondicionamiento para energizar los muelles se implementa 10 veces (una para cada muelle de Nitinol), y la etapa de acondicionamiento para energizar los ventiladores se implementa dos veces. El diseño incorpora elementos como conectores que permiten desmontar fácilmente la tarjeta y se agregan interruptores y leds que permitan controlar y vigilar el funcionamiento de la misma. Se incluye un regulador de voltaje 78-05 que permite garantizar un voltaje de alimentación a los potenciómetros de sensado y así evitar que variaciones de voltaje afecten la lectura. Además el diseño dispone de espacio para ubicar la tarjeta Arduino UNO con el propósito de construir un sistema mucho más integrado.

El diseño se realizó empleando EAGLE [44], un software para el diseño de PCB. Este programa cuenta con editor de esquemáticos que permite crear planos electrónicos, y luego mediante su editor de distribución tener una perspectiva 2D de la ubicación de cada componente en la tarjeta. Brinda además la posibilidad de crear impresos de 1 hasta 16 capas, cuenta con una amplia librería de componentes y los que no se encuentren pueden ser creados o modificar las existentes mediante su editor de librería [44]. El diseño esquemático se presenta a continuación.

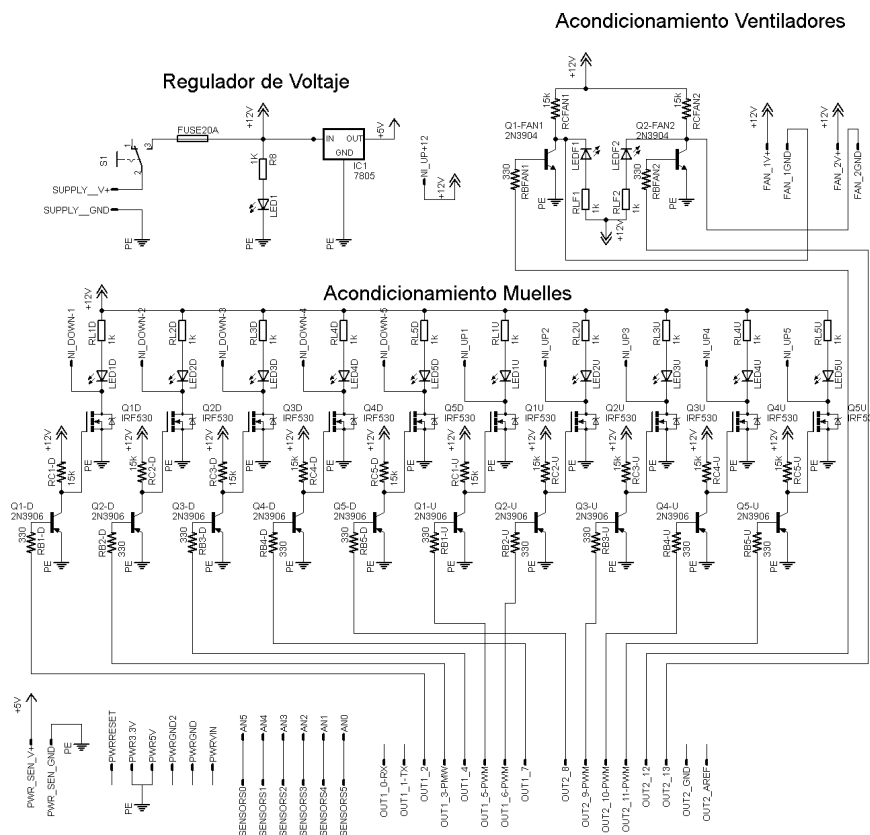


Figura 44. Diseño esquemático elaborado.

6.5.3. Fabricación de la tarjeta de acondicionamiento.

La construcción de la tarjeta impresa (Figura 45 (a)) se realizó empleando un método casero de fabricación de PCB. El proceso de fabricación se explica con mayor detalle en el ANEXO B.

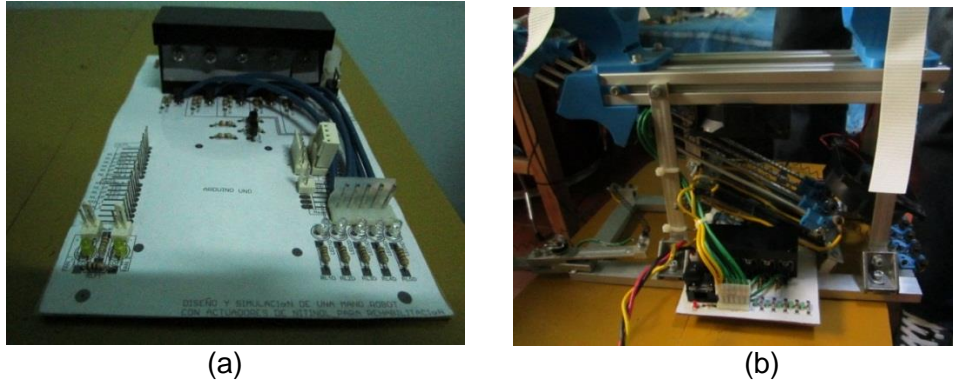


Figura 45. Tarjeta construida: independiente (a) e integrada (b)

Esta tarjeta fue sometida a verificaciones de funcionamiento para luego ser integrada con la estructura mecánica del sistema en el momento de su construcción (Figura 45 (b)).

6.5.3.1. Disipadores

El disipador es un elemento con alta conductividad térmica, que permite refrigerar dispositivos que presentan excesivo calentamiento. En nuestro caso se empleara un disipador de aluminio para refrigerar los 10 Mosfet que activan los muelles.

Los Mosfet presentan una resistencia interna variable, que se incrementa a medida que la temperatura en la juntura del material sube (Figura 46), por lo que es primordial mantenerlos refrigerados con el fin de evitar disipación extra de potencia.

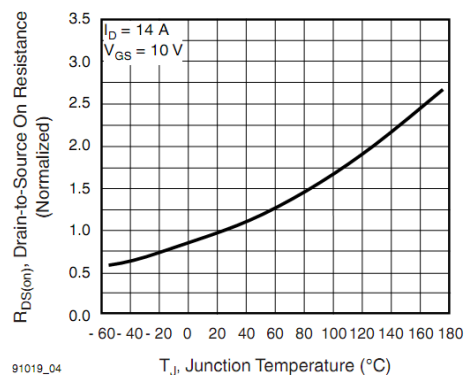


Figura 46. Grafica normalizada resistencia interna Vs temperatura.
Fuente: [45]

Los resultados de la Figura 46 se han obtenido sometiendo el mosfet a un voltaje de compuerta (VGS) de 10 voltios y una circulación de corriente de colector de 14 amperios,

además se debe tener en cuenta que los datos entregados por el fabricante corresponden a pruebas realizadas activando el mosfet durante periodos de microsegundos, y en este caso se requiere mantener energizado los muelles durante periodos de 10 a 14 segundos para realizar las fases de extensión y flexión, como consecuencia la temperatura y la resistencia interna se incrementaran.

Para evitar este inconveniente, y debido que el fabricante no proporciona información de la disipación o temperaturas alcanzadas por el Mosfet cuando se energiza durante largos periodos, se decide realizar tres experimentos adicionales. En el primero se energiza un Mosfet para activar un muelle de Nitinol durante un periodo de 14 segundos, a fin de determinar la temperatura máxima alcanzada. En el segundo, se determinará la temperatura alcanzada por el Mosfet emperando un disipador de 8X4X1 cm. En el último experimento se energizan 5 Mosfet, buscando imitar la situación en que el sistema tiene que extender o flexionar los 5 dedos al tiempo, con el fin de determinar si el disipador empleado es capaz de impedir un incremento excesivo de temperatura.

Como sensor de temperatura se empleó una termocupla tipo k, acoplada a un multímetro capaz de realizar mediciones de temperatura (Figura 47).



Figura 47. Multímetro con sensor de temperatura

Los resultados se presentan a continuación:

Tabla 13. Temperaturas en el IRF530 con y sin disipador.

Experimento	Corriente Mosfet (A)	Temperatura (°C)
1	4.1	60
2	4.1	30
3	3.6	35

Se hace evidente que al no emplear un disipador la temperatura se incrementa considerablemente, aunque esta se encuentre dentro de los límites permitidos por el fabricante. Se decide emplear el disipador, procurando prolongar la vida útil de los Mosfet.

6.6. Diseño del software

En una terapia de rehabilitación la valoración que se realiza del progreso de los pacientes es tan importante como los ejercicios que se ejecutan durante las sesiones. La razón es que de acuerdo al estado de cada paciente los fisioterapeutas programan los ejercicios a

desarrollar en la terapia, y posteriormente le indican al paciente cuál es la manera en que debe realizar cada ejercicio. Así cada terapia es diferente de la anterior y cada vez se exige más del paciente para que supere con éxito su discapacidad.

Por esta razón, si la terapia es asistida por un sistema mecánico de rehabilitación, este sistema debe permitir ajustar la terapia a las necesidades del paciente ofreciendo flexibilidad para modificar los ejercicios a realizar y poder supervisar la ejecución de los mismos.

6.6.1. Diseño de la interfaz de usuario

Para llevar a cabo la labor de control de la terapia se propone utilizar una aplicación cuya interfaz de usuario sea tan sencilla y clara como sea posible, presentando un orden de ejecución de la terapia lógico, haciendo fácil para el usuario deducir cómo debe programar una sesión y qué datos ingresar a la programación. Así, el diseño de la interfaz de usuario es muy importante dentro del presente proyecto por lo cual se desarrolló con la asesoría de fisioterapeutas experimentados en rehabilitación de pacientes afectados por ACV.

Realizando el diseño inicial de la interfaz se identificaron 2 etapas principales en las cuales se puede dividir una terapia. Una es la más evidente y consiste en ejecutar la terapia como tal, y la otra es decidir qué ejercicios se van a realizar durante la ejecución de la misma. Así, los componentes que debían presentarse al usuario en la interfaz durante la programación de los ejercicios debían permitirle escoger qué dedos ejercitar, en qué orden activarlos, hasta dónde extenderlos, y por cuánto tiempo o cuántas veces hacerlo, y durante la ejecución de la terapia simplemente interesaba que ésta se desarrollara sin inconvenientes de ningún tipo.

Una vez iniciado el proceso de diseño de esta primera interfaz, se identificó la necesidad de considerar como una nueva etapa de la terapia la fase de establecimiento del punto máximo a extender cada dedo, ya que se pensó que no sería práctico fijar estos valores cada vez que se deseara realizar un ejercicio, sino que sería más eficiente encontrar estos valores una sola vez y poder cargarlos en futuras sesiones. Entonces en la fase de programación de los ejercicios sólo se definiría qué secuencia ejecutar y por cuánto tiempo. A esta nueva etapa se le denominó etapa de diagnóstico, y por supuesto el rango de movimiento de los dedos almacenado en el archivo de cada paciente se puede modificar cuando sea necesario.

Sin embargo, con esta nueva etapa surgió también la necesidad de diferenciar si el usuario es nuevo o es antiguo, ya que si el usuario es nuevo, el diagnóstico de su rango de movimiento es obligatorio, y en caso de ser antiguo se puede omitir esta etapa para pasar directo a los ejercicios. Entonces se decidió incluir una etapa más para la terapia relacionada con el inicio de sesión de cada usuario, con lo cual se concluyó que no hacían falta más componentes en la interfaz de usuario y se procedió a realizar el diseño definitivo de la misma.

La interfaz de usuario se elaboró en Qt [46]. La gran ventaja de esta librería es su herramienta gráfica para edición de interfaces (Qt Designer), ya que permite visualizar el aspecto de la interfaz diseñada a medida que se edita. Para este proceso de diseño fue necesario adquirir conocimientos en las clases de Qt y los tipos de elementos que ofrece para creación de interfaces. Entonces, antes de continuar con la descripción del diseño de

la interfaz hace falta describir los aspectos básicos de esta librería y las clases más importantes utilizadas en esta aplicación:

- **QObject:** Clase base de todos los objetos de Qt, la cual permite la conexión de objetos por medio de *signals* y *slots*. Este mecanismo de conexión consiste en la emisión de señales cada vez que ocurre un evento en particular, las cuales vienen por defecto o se pueden crear o modificar a conveniencia, y llevan a cabo el llamado de funciones denominadas *slots* que pertenecen a otros objetos y también vienen por defecto con las clases y pueden modificarse o crearse según sea la aplicación. Además, QObject tiene como propiedad que los objetos siempre buscan organizarse en una estructura tipo árbol, lo cual facilita el llamado a destructores y el manejo de eventos cuando se instancia un objeto como hijo de otro garantizando que todas las modificaciones al padre se reflejan en los hijos [46].
- **QApplication:** Esta clase contiene el lazo principal de ejecución de una aplicación que contenga una interfaz gráfica de usuario (GUI) y permite el manejo de los eventos del sistema de ventanas, así se encarga de la inicialización y finalización de la aplicación. Independiente de cuántas ventanas posea la aplicación, es necesario al menos un objeto de esta clase para poder ejecutar una aplicación que contenga una GUI [46].
- **QPaintDevice:** Es la clase base para todos los objetos que pueden ser representados gráficamente. Contiene los métodos necesarios para refrescar lo mostrado en pantalla cada vez que se activan controles de la interfaz o se despliegan menús contextuales por ejemplo [46].
- **QWidget:** Es la clase base de todos los objetos de interfaz de usuario. Así, hereda de QObject y QPaintDevice. Es considerada el átomo de la interfaz de usuario ya que recibe eventos de teclado, ratón, y otros del sistema de ventanas, y pinta una representación de sí mismo en la pantalla. Así existen muchos tipos de *widget*, tal como ventanas, cuadros de diálogo, menús, y botones, entre otros [46].
- **QMainWindow:** Es una subclase de QWidget y sirve para crear ventanas principales para aplicaciones. Facilita la adición de barras de menús, barras de estado, barras de herramientas y la organización de *widgets* dentro de su espacio libre en diferentes *layouts* [46].
- **QDialog:** Es una subclase de QWidget que permite crear cuadros de diálogo que bloquean la interacción y eventos con la ventana principal mientras están activos. Ofrece botones por defecto con acciones asociadas a los mismos tal como aceptar, cancelar, abrir o guardar, entre otras [46].
- **Connect:** Es un método de la clase QObject que permite la conexión de *signals* y *slots*. Necesita que se le pasen 4 argumentos que son la señal y *slot* a conectar y a qué objeto pertenece cada una de estas [46].
- **Exec:** Es un método de la clase QApplication que inicia la ejecución del lazo principal de la aplicación en el que se permite mostrar los elementos de la interfaz de usuario y la recepción de los eventos relacionados con la misma [46].
- **paintEvent:** Es un manejador de eventos de la clase QWidget llamado cada vez que se necesita actualizar el aspecto del *widget* en pantalla, tal como minimizar la ventana o cambiar una pestaña por ejemplo. Es llamado automáticamente cada vez que el usuario interactúa con la interfaz o por código cuando se requiere actualización de un cambio reflejado en el aspecto gráfico realizado en alguna función durante la ejecución [46].

A partir de estas clases es posible construir una interfaz completa instanciando subclases de las mismas. En la sección 2 del Anexo E de este documento se describe el proceso de creación de una aplicación de Qt usando el asistente de configuración de proyectos para Visual Studio 2008 Express, y cómo se editan las interfaces de usuario con QtDesigner.

La clase principal de la aplicación es una instancia de QMainWindow, denominada "ventana1". Así en el archivo de código fuente que contiene el método main se instancia un objeto de la clase QApplication y uno de la clase principal. Primero se llama al método show() de "ventana1" para mostrar la interfaz, con lo cual se inicializan todos sus componentes, y a continuación se llama al método exec() del objeto QApplication para iniciar el manejo de eventos. El método show() simplemente crea los elementos de la interfaz y los organiza según la configuración definida en QtDesigner, ya que al iniciar la aplicación ninguno de estos existe. Para esto, cada vez que se compila el proyecto se genera automáticamente un archivo de cabecera correspondiente al diseño realizado en este editor gráfico, y por defecto, en el constructor de la clase principal se instancia un objeto que representa la interfaz. Hay que mencionar que cada vez que se insertaron elementos en la interfaz, se realizó la conexión entre las señales emitidas por los *widgets* de la misma al interactuar con el usuario y los *slots* por defecto o personalizados, donde se implementaron las acciones deseadas. Esta conexión se lleva cabo en el constructor de la clase principal por medio del método connect() de la clase QObject.

Lo primero que debía realizarse dentro de la ventana principal de la aplicación era dividir la interfaz en las 4 etapas definidas para una sesión de rehabilitación, para lo cual se consultaron los distintos tipos de agrupamiento de elementos gráficos ofrecidos por Qt. Para este caso se optó por la clase QTabWidget, que permite agrupar por pestañas los elementos de la interfaz, haciendo posible que en un mismo espacio de trabajo se puedan mostrar sólo los elementos que se necesita en determinado instante de la ejecución. Estas pestañas ocupan todo el espacio disponible de la ventana principal, excepto el espacio asignado para la barra de menús y el marco de la misma (Figura 48).

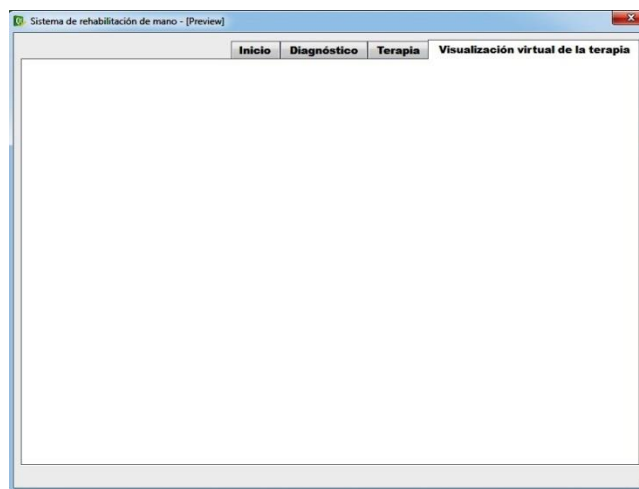


Figura 48. TabWidget usado en la aplicación.

Ya que la pestaña "Inicio" es la primera que el usuario observa al ejecutar la aplicación, en ella se muestra un mensaje de bienvenida (Figura 49) y en la barra de menús se ofrece al usuario las 2 posibilidades para iniciar la sesión las cuales son crear un paciente nuevo o

cargar uno tratado previamente. Además se muestra un botón para salir de la aplicación (Figura 50).

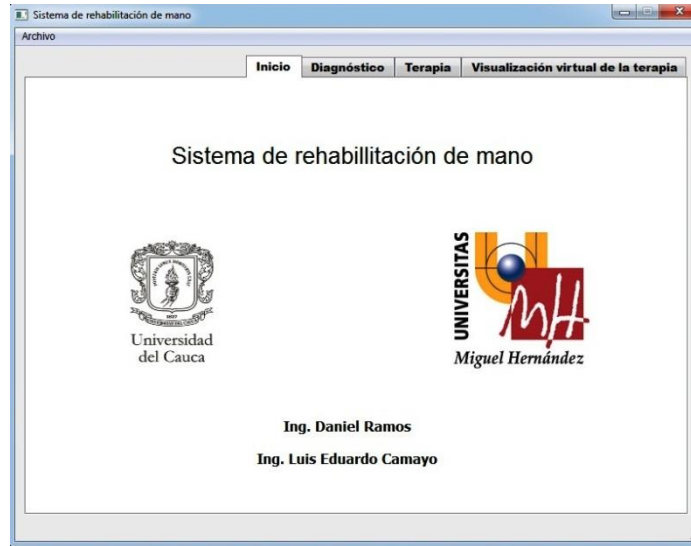


Figura 49. Pestaña de inicio.

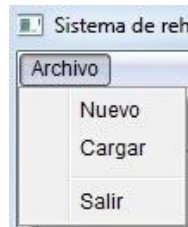


Figura 50. Opciones de inicio de sesión.

En caso que se cree un paciente nuevo, el mensaje de bienvenida desaparece para dar paso a un formulario (Figura 51), en el que se solicita ingresar el nombre del paciente, la patología que presenta y la fecha de la sesión, y es posible bien sea guardar el paciente o cancelar el proceso y volver a la pantalla de bienvenida. Si se decide guardar el paciente nuevo, se crea un archivo txt dentro de una carpeta del directorio de la aplicación destinada a almacenar los pacientes, siendo el nombre de este archivo el mismo ingresado en el campo "Nombre y apellido" (Figura 52).

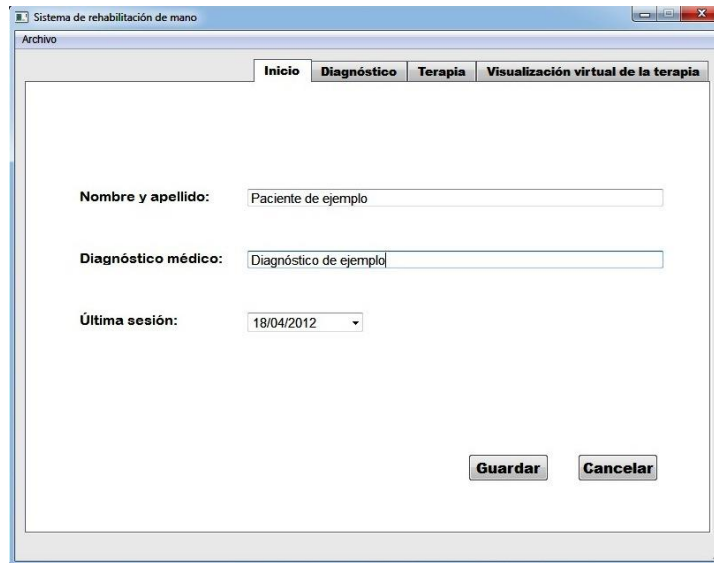


Figura 51. Ventana para crear un nuevo paciente.

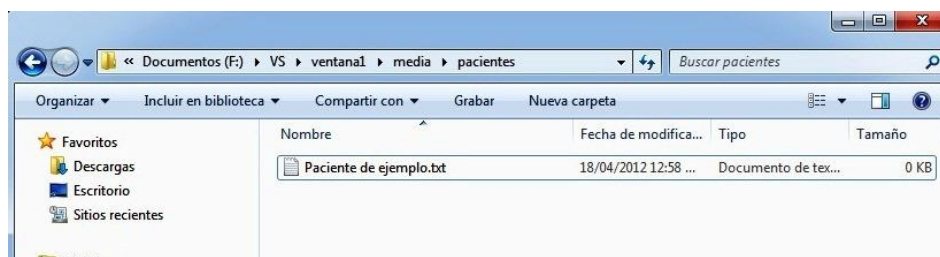


Figura 52. Creación de un archivo de texto para el nuevo paciente.

El manejo de los archivos txt se lleva cabo con la clase QFile, la cual permite acceder al contenido de los mismos, de modo que se puede leer la información que se necesite o descargar los datos generados en la aplicación al archivo, como es el caso de los nuevos pacientes. Con QFile también se pueden crear los archivos txt. Para crear el archivo se emplea el constructor de QFile pasándole como argumento la ruta y/o el nombre del archivo a crear, y se accede a éste con la función open() a la cual se le pasa como argumento el modo en que se abre el archivo, que puede ser WriteOnly si se desea escribir datos o ReadOnly si se quiere cargar los datos.

En caso de cargar el paciente, se muestra un cuadro de diálogo con un listado de los archivos almacenados en ese instante en el directorio de pacientes. Las posibilidades son cargar el paciente o cancelar el proceso y permanecer en la pantalla de bienvenida. En la Figura 53 se muestra este cuadro de diálogo pero en este caso la lista sólo contiene un paciente, creado para este ejemplo de uso de la aplicación.

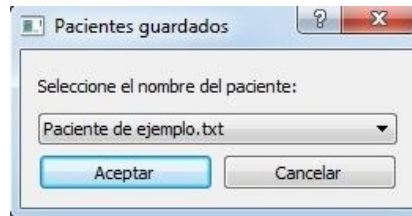


Figura 53. Cuadro de diálogo para cargar pacientes.

Los cuadros de diálogo son generados con la clase `QMessageBox`, la cual ofrece botones estándar que tienen funciones asociadas a las tareas más usadas en estos casos, tales como aceptar, cancelar, guardar, o abrir, por ejemplo. Así basta con definir el texto mostrado en el cuadro por medio de la función `setText()`, agregar los botones que se desee emplear con la función `setStandardButtons()`, y por último llamar al método `exec()` de esta clase para esperar el evento disparado cuando se presione uno de los botones. Además, se emplea la clase `QDir` para indicarle a la aplicación en qué directorios se deben buscar los archivos a cargar, que en este caso por ejemplo, consistiría en especificar la ruta del directorio que contiene los archivos de los pacientes, lo cual se hace pasando esta ruta como argumento al constructor de `QDir`. Es necesario usar esta clase porque permite controlar el contenido del directorio, más específicamente permite eliminar ficheros en los casos que se crea un nuevo paciente y si no se completa el diagnóstico, dejando el archivo incompleto, o generar en tiempo de ejecución listados con el contenido del directorio, lo cual se hace con la función `entryList()`.

Como se mencionó anteriormente, en caso que el paciente sea nuevo, es necesario que se llenen los datos de la pestaña de diagnóstico, los cuales sirven para caracterizar el perfil del usuario para la utilización de la máquina. Este perfil consiste en determinar la extensión máxima de cada dedo, la velocidad con que se extienden, y la flexión máxima que se permitirá a cada dedo, es decir, fijar el rango de movimiento de cada dedo. La extensión se fija en función del nivel de espasticidad que el paciente presenta o según el dolor que pueda experimentar al estirar sus dedos. La velocidad con que se realiza la extensión es variable ya que a pesar que el sistema no es rápido, los fisioterapeutas asesores manifestaron la necesidad de variar la rapidez con que se extendían los dedos para obtener distintos resultados en las terapias. El punto máximo permitido para flexionar los dedos se fija si se desea que la posición de partida de los ejercicios no sea con la mano casi cerrada sino con los dedos un poco extendidos, de modo que el cerebro se programa para evitar cerrar la mano en estado de reposo, característica recomendada por el equipo médico asesor. Para la caracterización por medio de estos 3 parámetros, no se ingresa el ángulo de inicio y final de cada articulación, ni la rapidez con que se extenderán los dedos, sino que se utilizan unas escalas con rangos relativos a los movimientos apreciables a simple vista por el fisioterapeuta, ya que es más práctico por ejemplo, escribir un número entre 1 y 20 para limitar la extensión de un dedo en lugar de escribir 3 ángulos entre 0° y 90° que se permitirá extender cada articulación del mismo. Entonces, se estableció una escala entre 1 y 20 para los valores de extensión, y 1 y 10 tanto para la velocidad como para la flexión, pero estos valores no tienen unidades.

Los elementos mostrados en la pestaña de diagnóstico son un *combo box* para seleccionar el dedo a diagnosticar y 3 *spin box* en los que se ingresan los valores de extensión, velocidad y flexión mencionados en el párrafo anterior para el dedo seleccionado. Además, ya que se debe verificar el rango de operación de cada dedo en el robot, se muestra también un botón que al ser presionado le dice a la tarjeta de control

qué dedo debe activar y con qué parámetros. Evidentemente, y en caso de sobrepasarse en la estimación de los 3 parámetros, se ha destinado un botón para detener el sistema de inmediato. Todos estos elementos se pueden apreciar en la Figura 54.

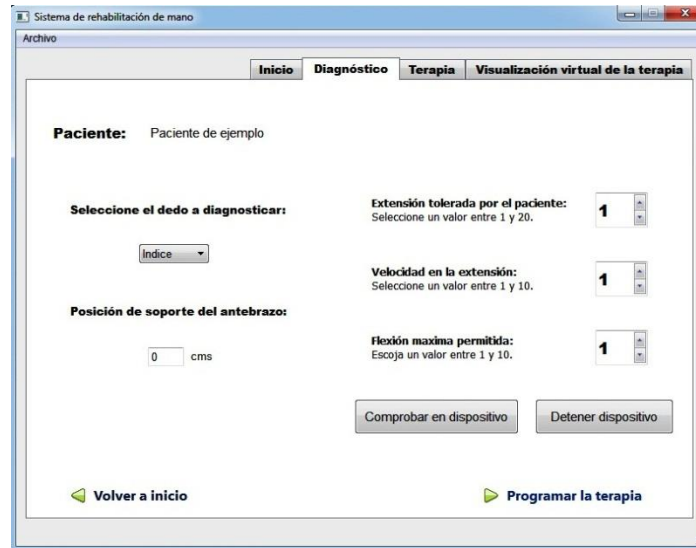


Figura 54. Pestaña de diagnóstico.

Una vez completado el diagnóstico, el usuario puede continuar a la fase de programación de los ejercicios presionando el botón “Programar la terapia” mostrado en la Figura 54.

En el caso donde el paciente es cargado, la información referente al diagnóstico del rango de movimiento de sus dedos se encuentra almacenada en el archivo que se carga. Entonces, tras seleccionar uno de los pacientes en el primer cuadro de diálogo mostrado al usuario, se muestra un segundo cuadro preguntando si desea seguir trabajando con los mismos valores o si desea modificarlos (Figura 55).

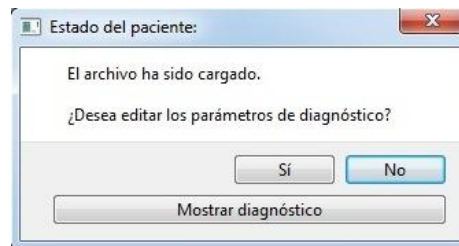


Figura 55. Cuadro para modificar el diagnóstico de los pacientes cargados.

En el caso de querer conservar los valores, se evita la etapa de diagnóstico y se pasa directo a la tercera pestaña, y en caso de desear modificar estos parámetros, se deja al usuario en la pestaña de diagnóstico para hacer la edición de los mismos. En este segundo cuadro de diálogo se ofrece la posibilidad de mirar los valores almacenados ya que es difícil que el usuario recuerde los 15 valores que caracterizan el rango de movimiento de sus dedos. Estos valores se muestran al hacer clic sobre el botón “Mostrar diagnóstico” del cuadro de diálogo desplegado. En la Figura 56 se pueden apreciar los datos almacenados en el archivo del paciente de ejemplo, por esta razón todos los valores están en 1.

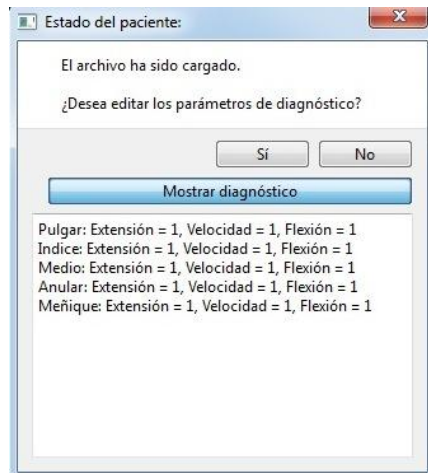


Figura 56. Visualización de los datos almacenados en el archivo de un paciente.

En la pestaña “terapia” se configuran los ejercicios que se van a realizar, para lo cual existen 2 opciones. Una es cargar un ejercicio de una base de datos incorporada en la aplicación y la otra es crear un ejercicio nuevo para ejecutarlo. Por esta razón se agruparon los elementos de la interfaz necesarios para cada caso en objetos de la clase QGroupBox, ya que permiten activar un *group box* y desactivar el otro haciendo ambos casos excluyentes entre sí (Figura 57).



Figura 57. Group Boxes para crear y cargar ejercicios.

En ambos casos, los parámetros que definen la terapia son pocos. Primero se define si se trabajará con varios o todos los dedos al tiempo o si en cambio se desea activarlos individualmente, como en una secuencia, y una vez establecido el tipo de activación para los dedos, es necesario definir qué dedos se rehabilitarán. En este punto se completa la información necesaria para guardar un ejercicio si este es nuevo, o en caso de ser cargado, esta es la información que se carga en la aplicación.

Para los ejercicios nuevos, se utilizan objetos de la clase QComboBox para seleccionar el tipo de secuencia de activación para los dedos y el orden de la activación, y objetos de la clase QCheckBox para seleccionar los dedos que se desea utilizar (Figura 58). Además, se ofrece la posibilidad de guardar los ejercicios nuevos en la base de datos para lo cual, una vez completados los parámetros necesarios para un ejercicio, se puede ingresar un nombre para el mismo en un objeto de la clase QLineEdit y confirmar la orden presionando el botón “Guardar” (Figura 59).



Figura 58. Selección del tipo de secuencia y del orden de activación de los dedos.



Figura 59. Guardar un ejercicio nuevo.

Cuando el ejercicio es cargado, por medio de un combo box se selecciona el ejercicio a cargar de un listado que contiene todos los que están almacenados en la carpeta de ejercicios, y en vista que es posible guardar aquellos creados en sesiones anteriores para tener acceso a éstos la próxima vez que se ejecute la aplicación, el listado de ejercicios se llena en tiempo de ejecución por medio de la función `entryList()` de la clase `QDir`, habiendo definido antes el directorio de ejercicios en el constructor de `QDir`. En la Figura 60 se puede ver una lista con los 3 ejercicios que hacen parte de la base de datos y el que se creó para el ejemplo.



Figura 60. Selección del ejercicio a ejecutar.

Para los ejercicios cargados también se ofrece la posibilidad de mirar en qué consiste cada ejercicio en un video que se puede reproducir al presionar el botón de vista previa, y detener la reproducción por medio del botón para ello (Figura 61). Para el manejo de los videos se utilizan las clases `VideoPlayer` y `MediaSource` del espacio de nombres *phonon*. La primera ofrece la funcionalidad básica de un reproductor de video y con la segunda se especifica la ruta del video. Evidentemente, no se contará con la vista previa de los nuevos ejercicios creados en las sesiones.



Figura 61. Vista previa de uno de los ejercicios de la base de datos.

Para poder iniciar la terapia hace falta configurar un parámetro más en esta tercera pestaña, que consiste en establecer la cantidad de repeticiones o el tiempo que durará la

misma. Esto se hace por medio de un *combo box* que permite escoger entre repeticiones o tiempo de duración, y un *spin box* para ingresar el valor deseado (Figura 62 (a)). Sólo hasta configurar este parámetro es posible cambiar a la última pestaña de la sesión, sin embargo, en caso de necesitar personalizar un poco más los ejercicios, se ofrece al usuario la posibilidad de fijar un tiempo en que los dedos permanecerán extendidos y el tiempo que permanecerán flexionados, para lo cual se utiliza un *combo box* para seleccionar el dedo, otro para escoger entre flexión y extensión y un *spin box* para ingresar el valor deseado. Sin embargo no es necesario ingresar estos datos para poder ejecutar la terapia, ya que solo es un rasgo adicional (Figura 62 (b)). En la Figura 63 se muestra la pestaña “terapia” completa.

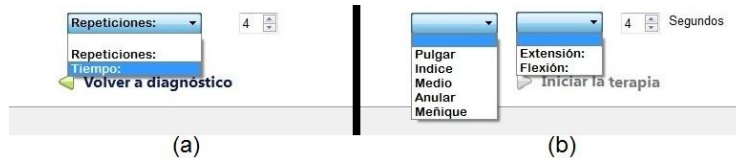


Figura 62. Selección de la duración de la terapia y del tiempo en extensión y flexión de los dedos.

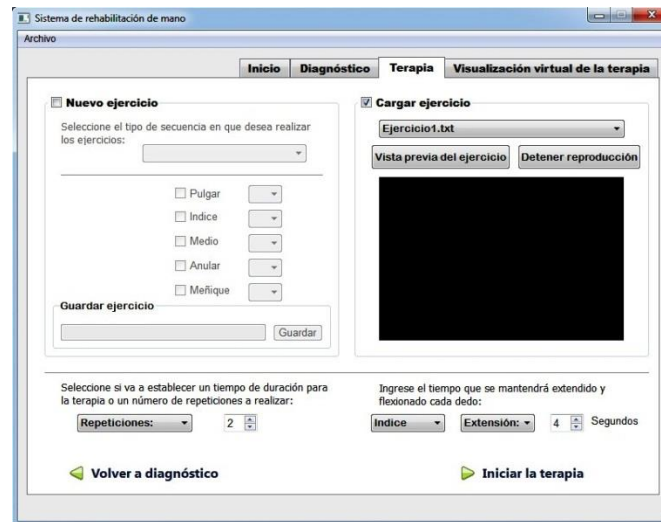


Figura 63. Pestaña para programar los ejercicios de la terapia.

Al presionar el botón “Iniciar terapia” (Figura 63) se descargan a la tarjeta de control las instrucciones para la ejecución del ejercicio. Así en la última etapa de la sesión el mecanismo empieza a actuar y no es posible programar nada más a menos que se desee detener el sistema. Entonces se propuso mostrar en la última pestaña de la interfaz un modelo virtual de la mano del paciente realizando los ejercicios, con el fin de quitar la atención del paciente del dispositivo que acciona su mano y concentrarla en observar los ejercicios realizados en una mano “limpia”, sin el sistema acoplado a ella, haciendo más agradable la experiencia del paciente durante la terapia (Figura 64). De acuerdo a los fisioterapeutas asesores, esta realimentación visual de una mano realizando la terapia sin nada montado sobre ella genera un impacto positivo en la rehabilitación de los pacientes, razón por la cual se trató que el modelo virtual mostrado fuera tan real y agradable como fuera posible, haciendo necesario complementar Qt con un motor de renderizado para imágenes 3D que permitiera conseguir la calidad mostrada en la Figura 64.

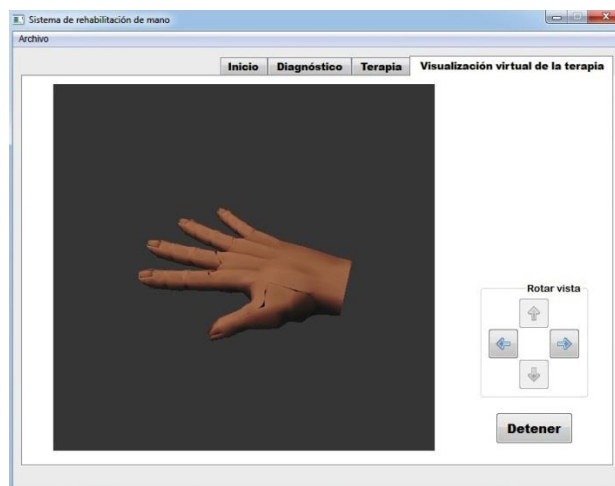


Figura 64. Visualización virtual de la terapia.

La mano virtual se renderiza con Ogre y se modeló en Blender, para lo cual hizo falta hacer una integración entre Ogre y Qt tal como se describe en la sección 4 del Anexo E de este documento, en la cual se puede apreciar que se implementa una clase en la que se anida la ventana de renderizado de Ogre dentro de un widget de Qt. En la aplicación creada para el presente proyecto de investigación, se denominó “ogrewidget” a la clase que contiene la ventana de renderizado, pero es instanciada como hija de “ventana1”, entonces, ya que en el constructor de la ventana principal se instancia la interfaz y todos sus elementos, también hace falta instanciar la ventana de renderizado, de modo que se tenga acceso a todo sus atributos y métodos desde el inicio de la aplicación, lo cual es muy importante porque en la clase “ogrewidget” se encuentran las funciones y variables involucradas en la comunicación serial entre la aplicación y el robot.

El proceso de modelado de la mano y de las animaciones con Ogre se describe en las secciones 6.6.2 y 6.6.3 de este documento.

6.6.2. Modelado 3D de la mano

Como se mencionó, la mano 3D mostrada en la interfaz de usuario se modeló en Blender. Ahora bien, para que esta mano pueda replicar los movimientos realizados por los pacientes durante la terapia, se debe hacer una lectura de la posición de las articulaciones de los dedos del paciente, lo cual se lleva a cabo en Ogre. Entonces en Blender se modela la malla que representa la mano con un esqueleto insertado en ella, el cual permitirá el movimiento de la mano en Ogre.

El primer paso del modelado es la construcción de la malla, la cual consiste en la mano con el aspecto y dimensiones que se desean para la visualización, tal como se muestra en la Figura 65 (a). Para lograr este resultado hace falta tener un manejo intermedio de Blender, el cual puede lograrse con la ayuda del Manual de Blender en Español [47], sin embargo la comunidad de usuarios de Blender comparte en internet modelos muy detallados y avanzados, con libertad para su descarga y utilización. Por ejemplo, la mano utilizada para la aplicación del presente proyecto fue descargada de Blend Swap [48].

Una vez terminado el modelado de la malla, se inserta un esqueleto en el modelo, cuyos huesos deben imitar los huesos de la mano humana necesarios para el movimiento de los dedos. Entonces para los dedos 2 al 5 se insertan huesos en las posiciones de las falanges y en el caso del pulgar, se inserta adicionalmente un hueso en la posición del primer metacarpiano. Para finalizar el esqueleto es necesario insertar otros huesos que sirven como referencia para variar la posición de los huesos de los dedos, los cuales se insertaron a lo largo de la palma hasta la muñeca (Figura 65 (b)). Recordemos que al proceso de construir una cadena de objetos que se puedan animar, tal como el esqueleto con sus huesos, se le denomina *rigging*.

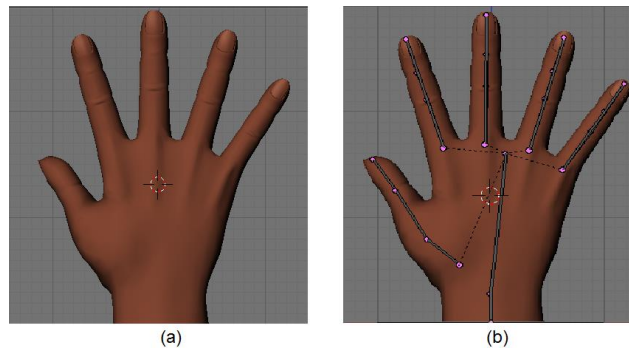


Figura 65. Malla y esqueleto modelados en Blender.
Fuente: [48]

El esqueleto se posiciona visualmente, ayudándose de todas las vistas que ofrece Blender. Cuando se ubican todos los huesos en la posición deseada, es necesario asociarlos con los vértices de la malla, ya que así se le dice a ésta cómo deformarse con el movimiento de cada hueso. A este proceso se le conoce como *skinning*. En el capítulo IX del Manual en Español de Blender se puede apreciar en detalle el procedimiento del *rigging* y el *skinning* [47].

Estas son las operaciones que deben realizarse en Blender respecto al modelado. El siguiente paso es exportar el modelo construido a Ogre, para lo cual se utiliza el script Ogre Meshes Exporter [49], que toma el modelo guardado por Blender en formato XML y extrae la malla (*.mesh), el material (*.material) y el esqueleto (*.skeleton) como archivos individuales.

En Blender se cambia a la ventana de *scripts* y en el menú desplegable se ingresa en *Export* y se selecciona *Ogre meshes* (Figura 66). En la ventana del exportador hay que configurar la ruta en la que se guardan los archivos resultantes y algunas características de materiales. En el campo “*Selected*” se muestra lo que se ha seleccionado para exportar, lo cual debe coincidir con los objetos usados en el modelo, tal como un cubo o una esfera. En las especificaciones de material hay que habilitar la opción “*Copy textures*” y “*Game engine materials*”, con lo cual el resultado de la exportación incorpora las texturas creadas para el modelo y los materiales utilizados y mostrados en el motor de juegos. Además se puede escoger si el eje y del modelo apunta hacia arriba como en el sistema de coordenadas de Ogre, o si se conserva apuntando hacia el frente como en el sistema de coordenadas de Blender (Figura 68). Se recomienda seleccionar “*Require materials*”, ya que habilitando esta opción se le dice a Blender que verifique si alguna parte de la malla se ha quedado sin un material asignado, lo cual produciría que la malla tuviera en esas zonas un color o aspecto diferente al resto de sí misma. Por último y de

gran importancia, es necesario seleccionar “OgreXMLConverter” para especificar qué *script* que se utilizará para exportar la malla (Figura 67).

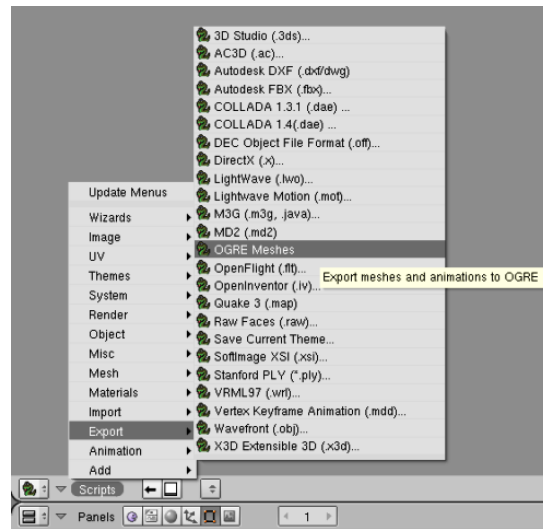


Figura 66. Llamado al exportador de mallas en Blender.

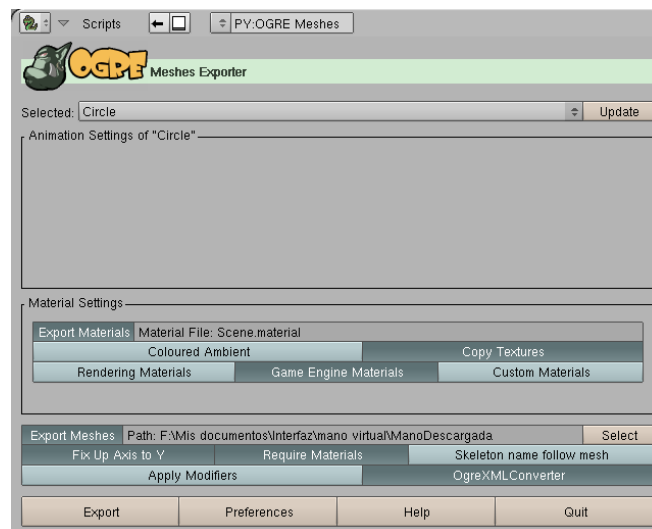


Figura 67. Ogre Meshes Exporter.

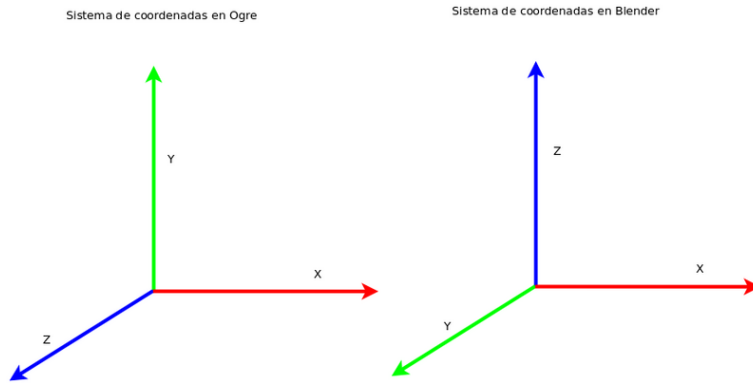


Figura 68. Sistema de coordenadas en Ogre y Blender.
Fuente: [33]

Por último, hace falta configurar las preferencias de exportación, haciendo click en el botón “*Preferences*”. Lo único que debe cambiarse se encuentra en la sección “*Location*”, donde se activa el botón “*Manual*” y se especifica la ruta en que fue instalado el script OgreXMLConverter, que por defecto se encuentra en el directorio en que fue instalado Ogre en el directorio bin/debug (Figura 69). Se guardan las preferencias haciendo click en “*OK*” y de vuelta en la ventana del exportador se finaliza el proceso de exportación haciendo click en el botón “*Export*”. Ahora bien, los archivos generados por el exportador de mallas quedan guardados en la ruta que se estableció antes, así para utilizarlos en la aplicación deben copiarse en las carpetas del proyecto en Ogre, lo cual se describe en la siguiente sección.

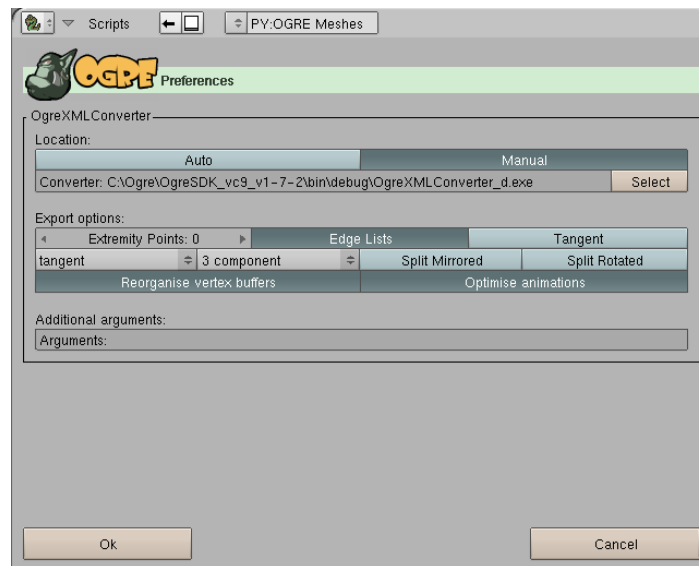


Figura 69. Configuración de las preferencias de exportación.

6.6.3. Animación del modelo 3D de la mano en Ogre

En el presente proyecto de investigación se utiliza Visual Studio 2008 como entorno de desarrollo para la aplicación. Una ventaja de utilizar este compilador es que existe un asistente de configuración para proyectos de Ogre que facilita y agiliza en gran medida el proceso de creación de una aplicación básica. La integración de Ogre con Visual Studio

es automática y muy simple por medio del asistente, sin embargo para crear un proyecto hace falta seguir una serie de pasos descritos en la sección 1 del Anexo E de este documento. Una vez se ha compilado y ejecutado el proyecto con éxito, se cuenta con un directorio con el código fuente, los archivos de cabecera y todos los recursos que utiliza el proyecto tal como imágenes, mallas y bases de datos, entre otras. La aplicación generada con el asistente se utilizó para el desarrollo de la fase de animación de la mano y comunicación serial con la tarjeta de acondicionamiento y control, pero una vez se logró un correcto funcionamiento de estas características, se continuó utilizando la configuración mencionada en la sección 3 del Anexo E de este documento, en el cual se menciona cómo crear una aplicación de Ogre sin el asistente y con el mínimo de componentes posible para que funcione.

El asistente de configuración carga por defecto la cabeza de un ogro, y la escena no contiene más que una luz. Esta cabeza de ogro se encuentra en el directorio del proyecto en la carpeta `media\models`, y es un archivo con extensión `.mesh`. En esta carpeta `media` también se encuentran los archivos de textura, material, partículas y fuentes utilizadas en el proyecto. Ahora bien, hace falta conocer algunas clases de Ogre que permiten la manipulación de la malla y los métodos de los que hace uso para visualizar movimiento en la ventana de renderizado.

- **Root:** Es el punto de inicio de todas las aplicaciones de Ogre. Si no se instancia un objeto de esta clase no se puede acceder a ninguna función de Ogre, así hace las veces de conexión entre la librería y las aplicaciones creadas por los usuarios [50].
- **SceneManager:** Gestiona la organización y el renderizado de una escena, para lo cual genera un árbol de renderizado que consiste en la jerarquía de nodos creada por el usuario al configurar su escena. Esta jerarquía también depende del SceneManager implementado [50].
- **Entity:** Es una clase con la que se define una instancia de un objeto basado en una malla [50]. Son los objetos que se pueden renderizar, así una luz no podría ser una entidad, por ejemplo.
- **SceneNode:** No son renderizables, por lo cual siempre se asocia una entidad a uno de estos nodos, ya que estos guardan la posición y orientación de todos los objetos asociados a ellos. Por esta razón siempre es necesario crear objetos de ambas clases para poder renderizar una imagen en pantalla [33].
- **SkeletonInstance:** Sirve para representar una instancia de la clase `Skeleton`, y es usado normalmente cuando no se crea el esqueleto en Ogre sino que se carga uno modelado en otro entorno [50].
- **Bone:** Con esta clase se definen los huesos de un esqueleto [50]. Al igual que en Blender, guardan relación con los vértices de la malla para caracterizar la deformación de la misma.
- **CreateScene:** Método implementado por el asistente de configuración en el cual se definen los componentes básicos de la escena, como son los nodos, las entidades, los esqueletos y las luces. Como se mencionó, por defecto el asistente sólo define una entidad con la cabeza del ogro y la asocia con un nodo de escena, y crea una luz y una cámara.
- **FrameListener:** Un *frame listener* es una clase que permite ser registrada para recibir notificaciones antes de que un *frame*¹⁶ sea renderizado en pantalla [33].
- **CreateFrameListener:** Función con la que se crea e inicializa un *frame listener*.

¹⁶ Un *frame* es un cuadro o una imagen individual dentro de una animación. La sucesión continua de varios *frames* produce una sensación de movimiento.

- `FrameRenderingQueued`: Función de la clase `FrameListener` con la cual se actualiza cada *frame*, y es llamada justo antes que los objetivos de renderizado se hayan actualizado [33].
- `ProcessUnbufferedInput`: Función con la que se escuchan los eventos relacionados con los métodos de la librería OIS en cada *frame*, tal como el movimiento del ratón o el presionar alguna tecla [33].

En el directorio en que se ha generado el proyecto se debe localizar la carpeta con nombre *media*, y pegar en ella los archivos generados con el exportador de mallas de Blender. Los archivos `.mesh` y `.skeleton` se pegan en *media/models*, el `.material` en *media/materials/scripts* y el `.png` con la textura en *media/materials/textures*. Una vez hecho esto es necesario modificar la función `createScene` en el archivo de código fuente generado por el asistente de configuración la siguiente línea:

```
Ogre::Entity* ogreHead = mSceneMgr->createEntity ("Head",
"ogrehead.mesh");
```

Por:

```
Ogre::Entity* mano = mSceneMgr->createEntity("Mano", "Circle.mesh");
```

`Circle.mesh` es el nombre de la malla importada desde Blender.

Una vez compilada y ejecutada la aplicación, en la pantalla de renderizado ya no se muestra la cabeza del ogro sino la malla elaborada en Blender (Figura 70).



Figura 70. Renderizado de la malla por defecto y la importada en Ogre.

Como se ha mencionado, la animación de la malla se facilita mucho utilizando un esqueleto asociado a la misma. Por lo tanto, una vez verificado que la malla se renderiza adecuadamente, el siguiente paso en la construcción de la aplicación consiste en instanciar el esqueleto exportado desde Blender y cada uno de los huesos que intervienen en el movimiento. Esto se realiza en la función `createScene` una vez realizada la asociación entre la entidad y el nodo de escena. Primero se crea un objeto de la clase `SkeletonInstance`, llamado *skel* en el ejemplo, y se llena con el esqueleto del modelo creado en Blender, lo cual se realiza con el método `getSkeleton` de la clase `SceneNode`. Una vez definido el esqueleto, se puede definir cada hueso, lo cual en el ejemplo sólo se hace para la falange proximal del dedo índice, instanciada con el nombre `index1`. Entonces se crea un objeto de la clase `Bone` y se llena con un hueso del objeto *skel*, usando el método `getBone` de la clase `SkeletonInstance`, para lo cual se necesita

conocer el nombre que se dio al hueso cuando se creó el modelo. A continuación se muestra el código para la carga del hueso de una falange del dedo índice.

```
Ogre::SkeletonInstance* skel = mano->getSkeleton();
Ogre::Bone* index1 = skel->getBone("Bone.002");
```

Para probar la deformación de la malla se simuló la entrada de los sensores con el teclado, con el fin de realizar ajustes al código de la animación antes de encargarse del código para la adquisición. Para utilizar como dispositivo de entrada el teclado se utiliza la función `processUnbufferedInput`, la cual recibe como argumento la estructura `frameEvent`, la cual contiene información sobre los eventos de cada frame. El método para emplear las teclas como entrada es `isKeyDown` de la clase `Keyboard` del espacio de nombres `OIS`, el cual registra un evento cuando una tecla es presionada. En el ejemplo, `mKeyboard` es el nombre del objeto. Así mismo, la sintaxis correcta para definir las teclas que se desea usar se obtiene del espacio de nombres `OIS`.

Para mover los dedos se implementó la función `mover_indice`, en la cual se obtiene la orientación del hueso teniendo en cuenta la orientación heredada con el método `getDerivedOrientation` de la clase `Bone`, y se guarda en un `quaternion`¹⁷ del cual se extrae un vector perpendicular al eje longitudinal del hueso por medio de la función `zAxis` de la clase `Quaternion`, para entonces aplicar la rotación deseada con el método `rotate` de la clase `Bone`, cuyos parámetros son el vector alrededor del que se gira, el ángulo de giro en grados, y el espacio de coordenadas respecto del que se hace la rotación, que en este caso es el global. A continuación se presenta el código usado:

```
void mover_indice(Ogre::Real angulo_indice)
{
Ogre::Quaternion q = index1->_getDerivedOrientation();
Ogre::Vector3 v_normal = q.zAxis();
index1->rotate(v_normal, Ogre::Degree(angulo_indice), nodo->TS_WORLD);
}
```

Además hay que decirle al *listener* que actualice el estado del esqueleto en cada frame con la función `_updateTransforms` de la clase `SkeletonInstance`, así:

```
skel->_updateTransforms();
```

Para la mano completa hace falta instanciar el resto de huesos de la mano y modificar la función para mover los dedos tal que se pueda aplicar la rotación deseada sobre cada uno de ellos.

Ahora hay que prestar atención en la lectura de los datos entregados por los sensores del sistema para mover cada dedo. Sería ideal montar un sensor en cada articulación de la mano para conocer todas las posiciones angulares, sin embargo esto resulta un poco complicado por el espacio del que se dispone alrededor de la mano, y más aun considerando que es una mano espástica. Por lo cual se plantea utilizar sólo un sensor por cada dedo, el cual mide el desplazamiento que ha realizado un alambre de Nitinol, y a

¹⁷ Un quaternion se puede expresar como un vector de 4 dimensiones de la forma $q = \{x, y, z, w\}$, donde los componentes x, y, z corresponden a los ejes del objeto y w la rotación a aplicar. El valor del quaternion se obtiene de: $q = \{ \sin(a/2)*nx, \sin(a/2)*ny, \sin(a/2)*nz, \cos(a/2) \}$, donde a es el ángulo de rotación.

partir de este dato determinar los ángulos de las 3 articulaciones de cada dedo correspondientes a las lecturas de cada sensor, con la ayuda de una tabla de valores construida a partir de experimentación.

Para consignar los valores de la tabla se decidió utilizar el lenguaje xml, ya que facilita la estructuración de la información en nodos para cada dedo y atributos con los ángulos de los dedos y los valores leídos en la adquisición. Un ejemplo de cómo se almacena la información se presenta a continuación:

```
<?xml version="1.0"?>
<raiz>
  <indice th3="-90" th2="-90" th1="-90" daq="10"/>
  <indice th3="-60" th2="-60" th1="-70" daq="20"/>
  <indice th3="-30" th2="-30" th1="-45" daq="30"/>
  <indice th3="-5" th2="-5" th1="-20" daq="40"/>
  <indice th3="0" th2="0" th1="0" daq="50"/>
</raiz>
```

Así con un bucle *for* en Ogre se busca entre los nodos de nombre índice aquel cuyo atributo "daq" coincida con la lectura del sensor en ese instante, y en caso de encontrar aquel nodo, se pasan los valores de los ángulos a la función encargada del movimiento del dedo. Entonces es posible guardar en un solo documento xml los ángulos y las lecturas para todos los dedos, ya que se los puede diferenciar al nombrar los nodos.

La librería para lectura de documentos xml usada es pugixml [51], escrita en C++, libre y cuya documentación es precisa y detalla cada función y clase de la librería. Para utilizarla sólo hay que agregar el archivo pugixml.hpp como archivo de cabecera en el proyecto. Las clases y métodos de esta librería que fueron utilizados en el proyecto son [51]:

- xml_document: Clase en la que se cargan los archivos a leer.
- xml_parse_result: Clase para verificar errores al buscar o cargar el archivo xml.
- xml_node: Clase para instanciar los nodos del documento. Con la función child de la clase xml_document se llena el contenido de los objetos xml_node.
- next_sibling: Función para cambiar de nodo cuando se leen los documentos.

En la sección 6.6.4, se describe el procedimiento empleado para transmitir desde la tarjeta Arduino UNO los datos provenientes de los potenciómetros.

6.6.4. Programación tarjeta Arduino UNO

La programación de la tarjeta se realizó empleando el software de programación Arduino, una herramienta gratuita y sencilla de emplear [41].

La programación se ha modularizado en procedimientos y funciones, así se logra una mejor comprensión de la secuencia de operación y además brinda la posibilidad de realizar mantenimiento, correcciones y actualizaciones de una forma rápida.

En la Tabla 14 se presentan los procedimientos y funciones implementadas, así como una descripción de su propósito.

Tabla 14. Procedimientos y funciones implementados en Arduino UNO.

Función /Procedimiento	Descripción
Void setup()	Permite inicializar las variables en un valor deseado, además de configurar la comunicación serial. Es el primer método que se ejecuta y lo hace una vez al energiza la tarjeta.
Void loop()	Este método es obligatorio implementarlo y se encarga de ejecutar constantemente las instrucciones definidas aquí.
void reiniciar_sistema()	Método que energiza los muelles de flexión para ubicar el sistema en un punto de partida.
void activar(modos, dedo)	Método que energiza un dedo, recibe como parámetro la forma como se activará el dedo ('e' = extensión o 'f' = flexión) y el número del dedo a activar 0 (pulgar) a 4 (meñique).
void activar(modos, pu, in, med, an, me, ven)	Permite activar más de dos dedos, recibe como parámetro la forma como se activarán los dedos ('e' = extensión o 'f' = flexión) y se especifica para cada dedo si se desea activar o no colocando 'a' o 'd'.
void control()	Después de activar uno o varios dedos se debe llamar a este método que se encarga de controlar cuando los dedos han alcanzado la extensión o flexión, definida por el usuario. Además este método se encarga de realizar las lecturas de las entradas analógicas que reciben los datos provenientes de los potenciómetros del sistema.
void apagar_de(d)	Desactiva la alimentación de los muelles que extienden y flexionan un dedo. Recibe como parámetro el dedo a desactivar.
void apagar_ven(ven)	Desactiva un ventilador en particular. Recibe como parámetro el número del ventilador a desactivar. 1 para ventilador muelles extensión, 2 ventilador muelles flexión, 0 apaga ambos.
void fase_1(d)	Esta fase se ejecuta cuando el usuario está configurando un dedo en la fase de acondicionamiento de la interfaz. Su función es activar un dedo a la velocidad determinada, llevarlo a la extensión máxima definida, y luego flexionarlo al valor mínimo definido por el usuario.
void fase_2()	Este procedimiento se encarga de ejecutar la terapia programada por el usuario, hace uso de los métodos activar, control y apagar para realizar la secuencia de activación de los dedos que ha definido el usuario.
void rx(c, d, _de, objetivo);	Esta función se encarga de recibir los datos transmitidos desde la interfaz, los parámetros c y d indican el número de iteraciones que debe realizar

	este método para recibir los datos, valores que cambian dependiendo de la fase que se encuentre especificada en el parámetro objetivo.
void desactivar()	Este método es llamado cada vez que se rebosan los <i>timers</i> que controlan el tiempo programado para mantener extendidos o flexionados los dedos, escribiendo en un vector el término 'd' indicando que se ha cumplido el tiempo.
void apagar_todo()	Método de seguridad llamado cuando el usuario presiona detener en la interfaz.
void tx_rx_fases()	Este método se encarga de constantemente estar transmitiendo información vital entre el sistema y la interfaz. Valores actualizados de los potenciómetros, tiempo transcurrido de la terapia y el estado de los botones de parada.
void acondicionamiento()	Este método se encarga de acondicionar toda la información recibida en el método de recepción de datos, fijando los valores rangos apropiados para su utilización.
boolean duracion_sesion(time_now)	Función que retorna verdadero si el número de repeticiones o el tiempo de terapia definida por el usuario han sido alcanzado. En caso contrario retorna falso.

6.6.5. Control del sistema

No fue necesario emplear una estrategia de control avanzada. La razón se fundamenta en la forma de operación simple de los muelles de Nitinol que al presentar características de comportamiento de memoria simple permite un control sencillo como se explicó anteriormente, por lo que el sistema de control empleado fue todo o nada.

En este tipo de control solo se presentan dos salidas de control ($u(t)$), una al 100% o activado y otra al 0% o desactivado. Las cuales conmutan dependiendo si la señal de error $e(t)$ es positiva o negativa.

$$u(t) = U1, \text{ para } e(t) > 0$$

$$u(t) = U2, \text{ para } e(t) < 0$$

El esquema de control se presenta a continuación

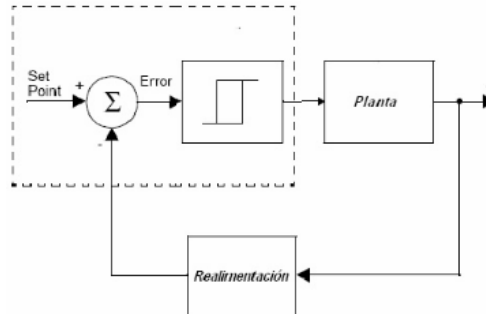


Figura 71. Esquema de control ON-OFF.
Fuente: [52]

Es primordial evitar un número excesivo de conmutaciones que deterioren rápidamente el actuador. Para solucionarlo se incluye una banda de histéresis que básicamente está definida como la diferencia de tiempos que existen entre las dos salidas de control U1 y U2.

Para el sistema elaborado la salida U1 no siempre estará al 100%, esta varía de acuerdo a la velocidad con que se desean extender los dedos, este parámetro es definido por el fisioterapeuta en la interfaz de usuario. El valor introducido se traduce en una variación del ciclo de trabajo de la señal PWM aplicada a los Mosfet que se encargan de energizar los muelles de Nitinol.

U2 corresponde a una desactivación completa de los Mosfet.

La retroalimentación es generada por las lecturas de las entradas analógicas cada 10 ms. A estas entradas están conectados los potenciómetros deslizables.

El set point es fijado por el usuario y corresponde a la extensión o flexión máxima que se desean alcanzar por el sistema. Este valor es fijado en la aplicación de usuario.

El error por lo tanto es la diferencia entre los valores de extensión o flexión y la lectura de las entradas analógicas.

La banda de histéresis asociada al comportamiento del sistema no es única, varía de acuerdo a los parámetros establecidos por el usuario. Así por ejemplo si se define una sesión de terapia en la que solo se desea ejercitar el dedo índice a una máxima velocidad, una curva de comportamiento generalizada para dos ciclos de operación sería la siguiente:

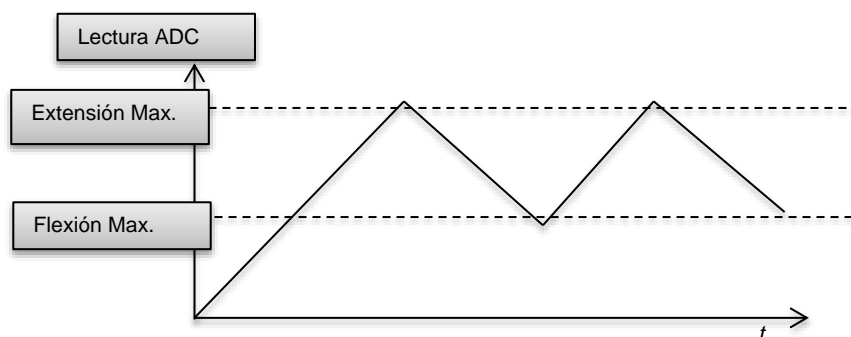


Figura 72. Banda de Histéresis generada para una configuración establecida.

Se determinó experimentalmente que en el peor de los casos cuando la banda de histéresis es mínima el tiempo de conmutación es alrededor de 5 segundos. Por lo que los Mosfet con tiempos de conmutación de microsegundos pueden operar fácilmente.

6.6.6. Integración hardware-Interfaz

Como se explicó anteriormente la interfaz se diseñó con el máximo de sencillez posible y considerando las sugerencias y recomendaciones de parte del Departamento de Fisioterapia de la Universidad del Cauca. Por lo tanto en esta etapa se busca traducir los requerimientos de usuario final en órdenes que pueda interpretar la tarjeta Arduino Uno y esta a su vez actué sobre el sistema implementado. La comunicación entre la interfaz y el hardware básicamente se hace a través de un paso sencillo de mensajes codificados en un vector en el que se almacena la información definida en cada fase de la interfaz. Este proceso de paso de mensajes entre la interfaz y la tarjeta Arduino se describe por medio del siguiente gráfico.

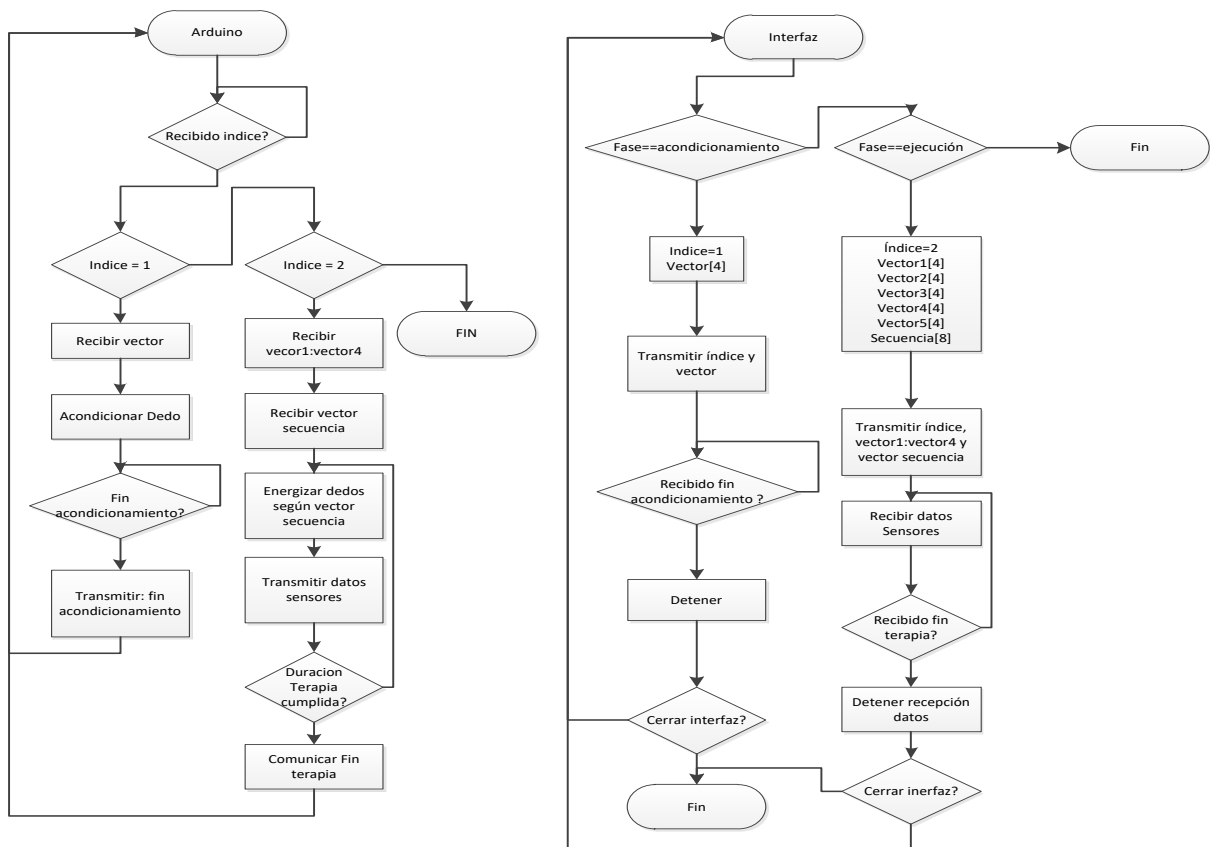


Figura 73. Diagrama de flujo representando el paso de mensajes entre la interfaz y la tarjeta Arduino UNO.

Inicialmente la tarjeta Arduino UNO, se encuentra en modo de espera de un índice transmitido por la interfaz que indica en qué etapa de configuración de la terapia se encuentra la interfaz. Cuando se transmite este índice, Arduino UNO crea las variables necesarias para almacenar la información que se dispone a recibir. Ahora si el índice señala que se está configurando la fase de diagnóstico, Arduino UNO sabe que recibirá un vector que contiene la información del dedo que quiere ser diagnosticado. Este vector contiene la información de la velocidad con que se quiere extender el dedo y los valores máximos de extensión y flexión para este. El vector es transmitido al presionar el botón “Comprobar dispositivo” de la pestaña “Diagnóstico” de la interfaz de usuario. Este procedimiento se repite hasta configurar todos los dedos.

Ahora si el índice señala que se configura la fase de terapia, se ha configurado Arduino UNO para que reciba dos vectores. En el primero la interfaz transmite la información correspondiente para cada dedo, y que es la misma información transmitida en la fase de diagnóstico. El segundo vector transmitido corresponde a la configuración de la terapia, en este vector se especifican los dedos a rehabilitar, el modo de trabajo ya sea energizar todos a la vez o secuencialmente, y la duración de la terapia en número de repeticiones o minutos.

Por ultimo al presionar el botón “Iniciar la terapia”, la tarjeta ejecuta la terapia programada, trasmitiendo hacia la interfaz los valores actuales de las posiciones de los dedos para poder ser imitados por el modelo virtual implementado en Ogre 3D. El proceso termina cuando se ha cumplido el tiempo de terapia o el número de repeticiones programadas, o si el usuario presiona algún momento el botón “Detener” de la cuarta pestaña de la interfaz, acto que ubica la interfaz en la fase configuración de terapia y reinicia automáticamente el sistema, retornando todos los muelles de Nitinol a la posición de flexión definida por el usuario. Quedando preparado para ejecutar una nueva sesión.

7. Pruebas y resultados

El dispositivo implementado se muestra de nuevo en la Figura 74, sin embargo, antes de experimentar con pacientes, se llevaron a cabo par de pruebas para evaluar el comportamiento de los componentes mecánicos y electrónicos.

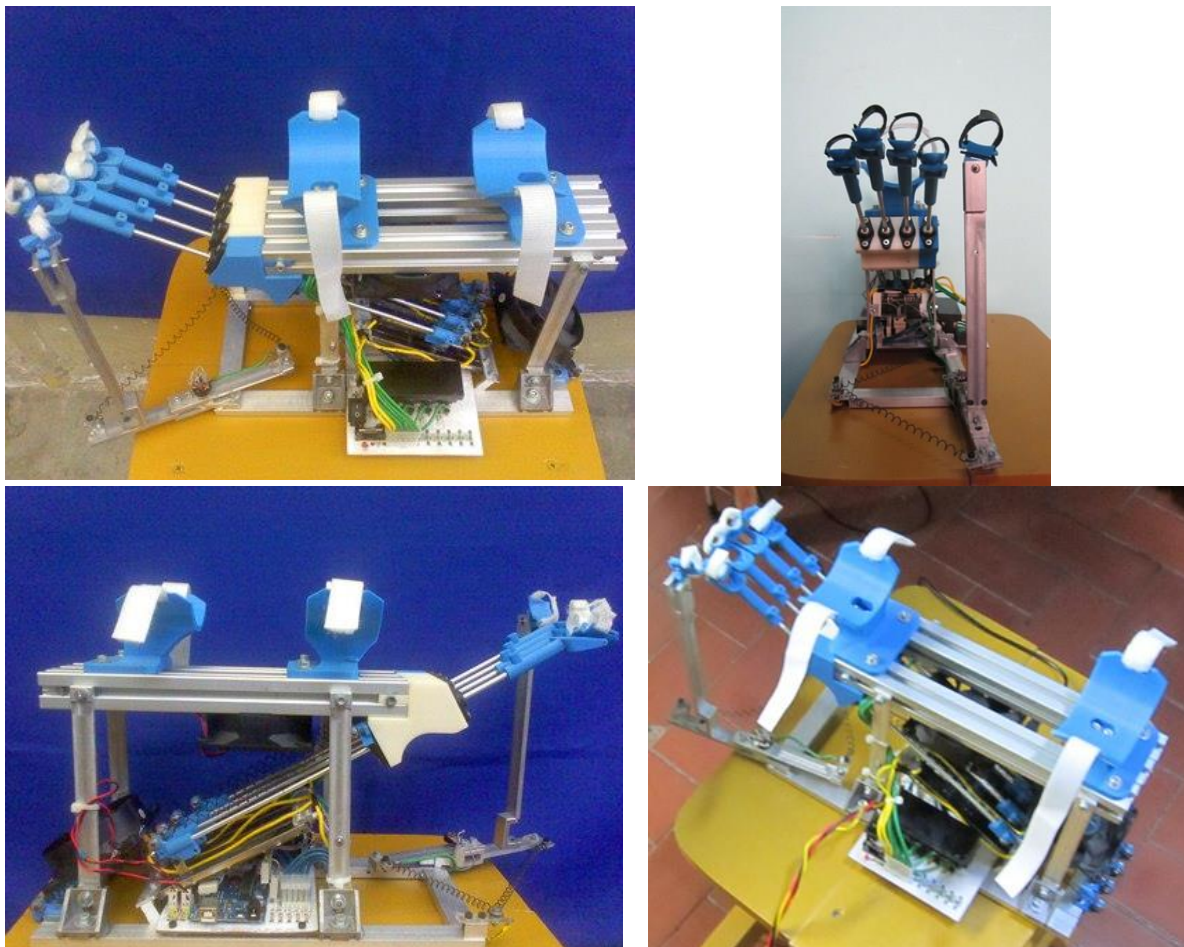


Figura 74. Dispositivo implementado.

La primera prueba realizada al sistema sin carga, permite corroborar que las órdenes especificadas en la interfaz son fielmente seguidas por el mecanismo. Además, se identifican y corrigen problemas que se podrían presentar al no hacer un uso adecuado de la interfaz. Como por ejemplo, al cerrarse accidentalmente la aplicación, el dispositivo continúe operando en vez de detenerse automáticamente como medida de seguridad. Además se verificó que en cualquier momento el sistema puede ser detenido inmediatamente al presionar el botón parar.

La segunda prueba permite determinar los tiempos empleados por cada dedo para realizar una extensión y flexión máxima permitida por el sistema energizando los muelles. Con el fin de no introducir error humano en la medición de los tiempos, la tarjeta Arduino es la encargada de registrar 10 mediciones en un vector para luego ser promediadas. Los resultados se presentan a continuación.

Tabla 15. Promedio de tiempos empleados por el sistema.

Dedo	Promedio Tiempos (s)	
	Extensión	Flexión
Pulgar	6	10
Índice	13	15
Medio	12	15
Anular	12	13
Meñique	12	13

Se puede apreciar en la Tabla 15 que los tiempos empleados por los dedos índice, medio anular y meñique son similares. Esto debido que se emplea la misma estructura mecánica de movimiento para todos ellos. Lo que no sucede con el dedo pulgar, en el que se empleó un mecanismo giratorio para realizar su extensión.

Además los tiempos registrados son similares a los resultados obtenidos en las sesiones de experimentación realizadas previamente.

Ahora se esperaría que los tiempos de extensión y flexión fueran los mismos, pero esto no sucede debido principalmente a la forma como se encuentran ubicados los ventiladores, ya que uno de ellos permite una mejor ventilación de los muelles que flexionan los dedos. Así cuando estos se encuentran calientes debido a una flexión del mecanismo, rápidamente pueden ser refrigerados para que el muelle de extensión lo deforme rápidamente. No fue posible ubicar apropiadamente el ventilador que refrigera los muelles que extienden los dedos debido al poco espacio disponible. Una versión futura promete atacar este inconveniente.

Además durante esta prueba se pudo determinar un consumo promedio de 4 A de corriente por muelle durante su extensión y flexión. Por lo que el consumo total de corriente del sistema cuando se estén rehabilitando todos los dedos a la vez es alrededor de 20 A. Este excesivo consumo de corriente es evidencia del principal inconveniente que presentan las aleaciones con memoria de forma.

Después de realizar las pruebas preliminares, el sistema fue probado en 2 personas. El paciente de prueba número 1 es un hombre de contextura delgada y estatura alta, sin problemas motores en sus manos. El paciente de prueba número 2 es una mujer de contextura media y mediana estatura, y sufrió un ACV hace 7 años, lo que le produjo una hemiparesia derecha. Las medidas de las manos de los pacientes se muestran en la Tabla 16.

Tabla 16. Medidas de las manos de los pacientes de prueba.

Sujeto de prueba	Longitud de la mano (cm)	Ancho de la mano (cm)	Separación entre los dedos (cm)
Paciente 1	20	9.5	2.5
Paciente 2	17	7.5	2

El paciente número 1 no presente ningún problema de movilidad en sus dedos con el fin de realizar una primera prueba de la forma en que el mecanismo se comportaba extendiendo los dedos. Además se esperaba que los dedos del paciente se extendieran de forma similar a la simulación, lo cual sucedió satisfactoriamente teniendo en cuenta

que las manos del paciente número 1 son más grandes que las de la mano modelada en Inventor y que en el modelo de simulación no se incluyó la articulación de la muñeca. En la Figura 75 se muestra una comparación de la situación real y la situación simulada.

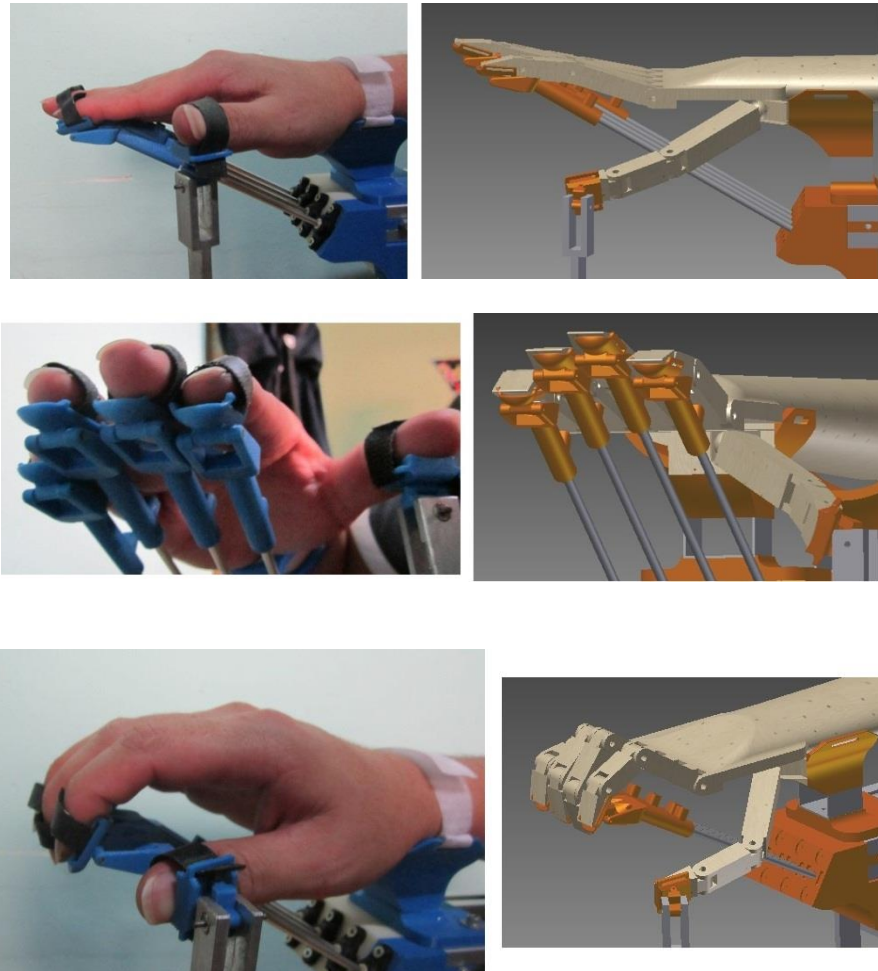


Figura 75. Comparación entre el mecanismo real y la simulación en Inventor.

Durante la fase del montaje de la mano del paciente sobre el mecanismo se observó que es un poco trabajosa la labor de aprisionamiento de los dedos con las cintas de velcro, pero una vez ajustada la mano al dispositivo, los soportes para el brazo brindan el apoyo necesario para que el paciente mantenga relajados sus músculos.

En esta prueba se le pidió al paciente que opusiera resistencia cuando el sistema extendiera sus dedos, y se pudo observar que los actuadores de Nitinol son capaces de aplicar la fuerza necesaria para vencer esta resistencia.

Con las observaciones realizadas en este experimento se ajustaron los valores de la tabla de conversión utilizada para obtener los 3 ángulos de cada dedo a partir de la lectura de los sensores.

En el caso del paciente número 2, se contó con la compañía de un fisioterapeuta de la Universidad del Cauca que preparó a la paciente para realizar la terapia con el sistema.

La paciente había suspendido sus terapias por inconvenientes personales, y recién estaba asistiendo de nuevo al programa de rehabilitación, de modo que sus músculos se habían endurecido un poco y manifestó haber perdido algo de movilidad desde que habían suspendido las terapias.

Los dedos más afectados de la paciente eran los 3 y 4. Su articulación más afectada en ambos dedos era la metacarpofalángica, de la cual sólo lograba una cuarta parte de la extensión máxima de la misma, en cambio sí podía extender por completo las articulaciones interfalángicas. Cuando se le pidió flexionar la muñeca se observó que, dentro de sus limitaciones, era capaz de extender los dedos. Sin embargo cuando se le pidió extenderla se pudo apreciar que no lograba extender ni un poco sus dedos. Respecto del control de sus movimientos, no podía realizar agarres con su mano afectada.

El fisioterapeuta realizó un trabajo de calentamiento del brazo completo de la paciente, incluyendo parte de la espalda, pero en ningún momento la paciente dio muestras de dolor ni incomodidad al movilizar sus miembros afectados, lo cual generó confianza para utilizar el robot ya que no se había probado antes en un paciente real.

Una vez finalizado el calentamiento, el fisioterapeuta dio el visto bueno para iniciar la terapia. Lo primero que se realizó fue el aseguramiento de los dedos al sistema (Figura 76), con lo cual se experimentó dificultad ya que los dedos de la paciente eran difíciles de manejar, y el poco espacio disponible para ajustar el velcro lo acentuó aún más. Sin embargo la tarea fue finalizada con éxito aunque tomara un poco más de tiempo que con el paciente número 1.



Figura 76. Montaje de la mano de la paciente sobre el sistema.

Cuando estuvieron fijos los dedos al dispositivo se ajustaron las bases de apoyo para el brazo de la paciente de acuerdo a sus dimensiones (Figura 77).

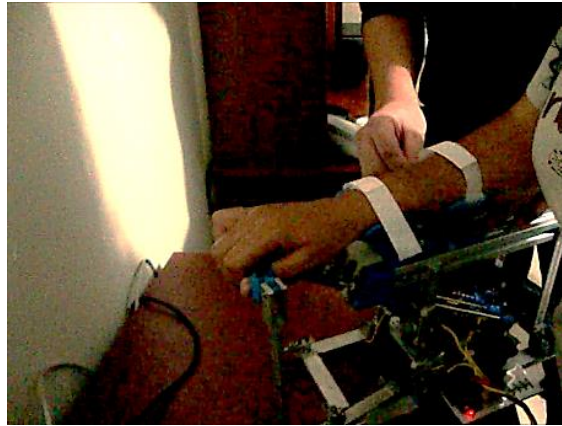


Figura 77. Ajuste de los soportes para el brazo de la paciente.

Completada la fase de montaje de la mano sobre el sistema se inició la interfaz de usuario creando un nuevo paciente (Figura 78). En la fase de diagnóstico se le explicaron al fisioterapeuta las escalas empleadas para llevar a cabo esta caracterización, así se ajustaron los valores para cada dedo bajo su supervisión. En este caso, ya que la paciente no manifestaba dolor ante el movimiento de sus miembros, el parámetro de extensión en el diagnóstico fue limitado por la dimensión de cada dedo. En la Figura 79 se muestran los valores de diagnóstico de la paciente.

Sistema de rehabilitación de mano

Datos del paciente

Inicio **Diagnóstico** Terapia Visualización virtual de la terapia

Nombre y apellido: Lisbeth

Diagnóstico médico: Hemiparesia derecha

Última sesión: 14/04/2012

Guardar Cancelar

Figura 78. Creación de un nuevo paciente para la sesión.

Pulgar: Extensión = 15, Velocidad = 10, Flexión = 10
Índice: Extensión = 17, Velocidad = 10, Flexión = 1
Medio: Extensión = 18, Velocidad = 10, Flexión = 1
Anular: Extensión = 18, Velocidad = 10, Flexión = 1
Meñique: Extensión = 10, Velocidad = 10, Flexión = 1

Figura 79. Valores diagnosticados para el rango de movimiento de la paciente.

Realizando el diagnóstico del rango de movimiento de los dedos de la paciente, se pudo observar que la articulación interfalángica distal y la articulación metacarpofalángica lograban extenderse en un 100 % de su rango de movimiento, sin embargo la articulación interfalángica proximal no logró la extensión máxima (Figura 80). El fisioterapeuta manifestó que esta situación no era problemática para esta paciente, sin embargo para casos en que la espasticidad fuera mayor, es posible que esta articulación sólo se extienda hasta un punto medio, generando un ejercicio de rehabilitación erróneo.



Figura 80. Extensión del dedo índice.

En el caso del dedo pulgar, hay que recordar que no se realiza flexión-extensión sino aducción-abducción, y observando el recorrido realizado por el pulgar, el fisioterapeuta dio el aval para incrementar el rango de movimiento permitido por los actuadores, ya que explicó que la máxima abducción permitida por el sistema en ese momento no era la máxima ideal manejada en las sesiones de rehabilitación. En la sesión experimental con la paciente número 2, el ángulo en abducción máximo alcanzado fue 45° (Figura 81), e idealmente debían alcanzarse 90° (Figura 82), aspecto que se ajustó para sesiones posteriores.

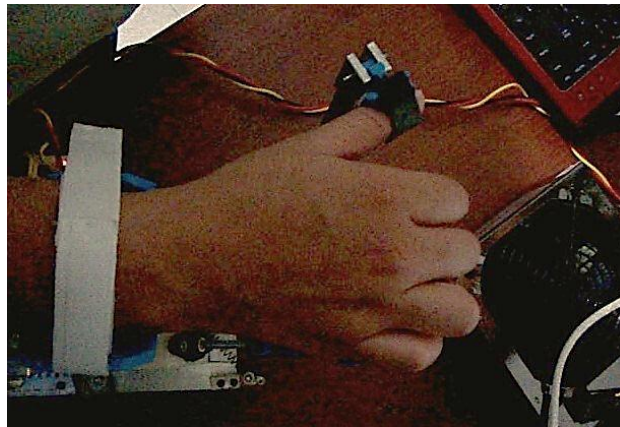


Figura 81. Abducción máxima del pulgar.



Figura 82. Comparación entre la abducción máxima del sistema y la abducción ideal del pulgar.

Finalizado el diagnóstico de la paciente, se procedió a realizar uno de los ejercicios de la pestaña terapia. Éste fue cargado de la base de datos de la aplicación, y consistía en abrir los dedos desde el 1 hasta el 5, y una vez extendido el quinto dedo, se procedía a cerrarlos en orden inverso (Figura 83). En esta secuencia se observó un comportamiento diferente en el cuarto dedo de la paciente en comparación con los otros dedos, que consistía en la tendencia de éste a separarse del eje longitudinal de la mano, lo cual es un efecto de la lesión padecida por la paciente. Lo que se observó fue que el cuarto dedo no llegaba a la misma postura de los otros dedos cuando se flexionaba, luciendo torcido y desalineado respecto de los otros. Sin embargo el fisioterapeuta asesor manifestó que este defecto puede corregirse con más sesiones de rehabilitación.



Figura 83. Secuencia ejecutada durante la terapia.

En cuanto al modelo virtual mostrado en la interfaz durante la terapia, la paciente manifestó que le parecía un rasgo importante ya que ella en particular sentía algo de inquietud mirando su mano manipulada por una máquina, y aseguró que tener una referencia que le mostrara cómo luciría su mano realizando los ejercicios sin el dispositivo sobre ella le daba motivación y ganas de trabajar en su rehabilitación. Además le aportaba más confianza para utilizar el sistema, ya que por momentos hacía de lado que

el robot estaba accionando su mano. En la Figura 84 se compara un movimiento realizado durante la terapia y su representación 3D.

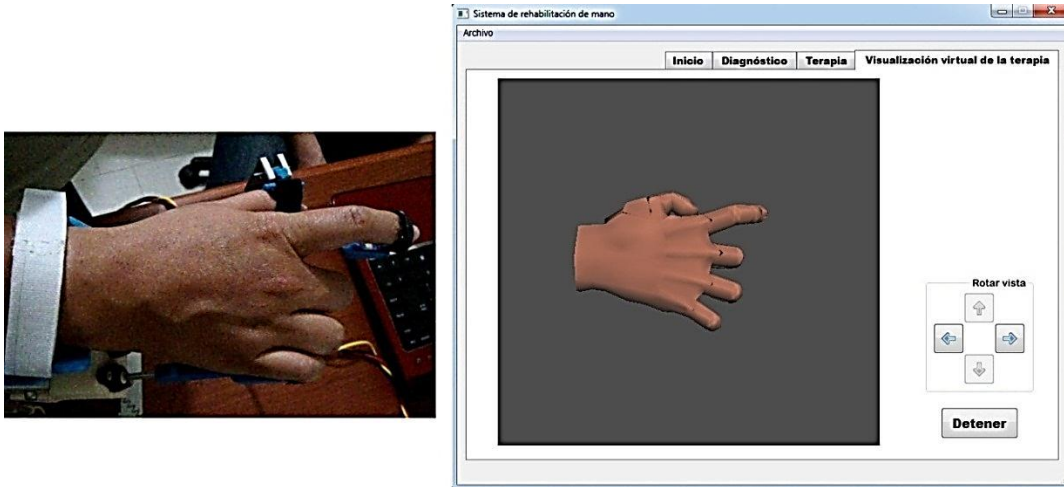


Figura 84. Representación virtual del movimiento de la mano del paciente.

7.1. Aspectos a mejorar

El sistema tiene muchos aspectos a mejorar. Estos se pudieron observar durante la construcción del dispositivo, en los experimentos realizados con los actuadores, y durante las sesiones experimentales finales, y se mencionan a continuación.

Refrigeración:

- Rediseñar la estructura mecánica de tal forma que el ventilador que refrigera los muelles de extensión se pueda ubicar más cerca de éstos, lo cual se puede conseguir rediseñando la pieza BC de tal forma que exista mayor separación entre ésta y la pieza SSA.
- En el diseño actual no se consideró emplear refrigeración del dedo índice, ya que experimentalmente se obtuvieron respuestas rápidas para extender y flexionar el dedo durante ciclos no mayores a 5 repeticiones. Por lo que es necesario incorporar un sistema de refrigeración si se desean obtener respuestas más rápidas y ciclos de operación superiores.

Seguridad:

- Un nuevo sistema incluiría tapas protectoras a los lados que permitan proteger al paciente de una posible quemadura con los muelles de Nitinol. Además de mejorar la estética y presentación del dispositivo.
- Incorporar un botón de parada de fácil acceso que permita interrumpir el paso de corriente al sistema.

Electrónica:

- Las dimensiones de la tarjeta de acondicionamiento fabricada se pueden reducir si se emplea un impreso de doble capa.
- El diseño actual se comunica empleando cable de datos USB, pero es posible establecer una comunicación inalámbrica que permita una mejor portabilidad del sistema.

Mecanismo:

- Es necesario modificar la mecánica del sistema de modo que los dedos no se muevan en línea recta, tal como lo hacen actualmente, sino que se considere que al flexionar y extender los dedos, involuntariamente se aducen y abducen, respectivamente. La primera situación se da en el sistema actual, pero la segunda no, tal como se puede apreciar en la Figura 84. Esta mejora aumentaría el espacio para maniobrar con los dedos e imitaría fielmente la biomecánica de la mano.
- Modificando la disposición de los actuadores del pulgar, o aumentando su cantidad, se podría lograr no sólo aducción-abducción de este dedo sino también flexión y extensión.
- Aumentar la separación entre las varillas que accionan los dedos para que se facilite el proceso de sujeción de los mismos al dispositivo, aunque también se podría cambiar este sistema de sujeción por otro más cómodo, tanto para el fisioterapeuta como para el paciente, ya que es necesario ajustar mucho el velcro para que los dedos queden bien asegurados.
- Realizar las modificaciones necesarias para que al extenderse los dedos, se logre extender por completo sus articulaciones interfalángicas proximales. Una manera sencilla para asistir esta extensión puede ser agregando una protrusión en la parte superior de las piezas ADx que vaya hacia las articulaciones interfalángicas proximales de cada dedo, ya que dimensionando adecuadamente esta nueva protrusión se empujaría hacia abajo las articulaciones en cuestión cuando el dedo esté extendido, y no interferiría con ningún movimiento cuando el dedo esté flexionado.
- Una mejora más radical podría ser rediseñar por completo el mecanismo para hacer posible realizar agarres con la mano, ya que el sistema usado para extender los dedos hace imposible acercar objetos a la palma de la mano para agarrarlos, e incluso no permite ni siquiera que el pulgar se oponga a los otros dedos, haciendo posible por ahora sólo el agarre lateral, mostrado en la Figura 3(f) de la subsección 2.1, "Características, funcionalidad y agarres", del presente documento.
- Rediseñar el mecanismo del dedo pulgar de tal forma que el sistema se pueda emplear también en pacientes con problemas de movilidad en la mano izquierda.

Piezas modeladas:

- En vista que el material de las piezas impresas resultó ser muy frágil cuando el grosor de la misma es reducido (3 o 4 mm), y que además aquellas piezas roscadas sólo pudieron ser atornilladas 3 veces antes de perder la rosca, una mejora considerable para el sistema consistiría en imprimir las piezas diseñadas para el mecanismo en metal o en una resina diferente, para que así se reduzca el riesgo de dañar aquellas piezas que requieren atornillarse o aquellas con poco grosor y que deben ser manipuladas constantemente, como lo son las piezas ADx.

Aplicación software:

- Hacer más agradable a la vista la escena renderizada en la última pestaña de la interfaz de usuario, bien sea cambiando la malla mostrada por una más detallada y realista, o también añadiendo un escenario de fondo que mejore aún más la experiencia del paciente mientras realiza la terapia.
- Agregar nuevos ejercicios a la base de datos que permitan programar secuencias para simular situaciones de la vida real, como puede ser marcar un número de teléfono, o incluir un modo de trabajo para realizar juegos en los que el paciente alterne los dedos empleados en el orden que el juego le diga, por ejemplo se puede simular que se toca un piano. Entonces habría que cargar escenarios para la escena renderizada que permitan visualizar estas situaciones.

8. Conclusiones

En el presente proyecto de investigación se realizó el diseño mecánico de un robot que permite extender y flexionar los dedos de una mano espástica. Este diseño presenta una característica interesante que simplifica su accionamiento, y consiste en que permite extender y flexionar las 3 articulaciones de los dedos de una persona a partir de un único movimiento lineal por dedo. Esta característica tiene gran importancia porque hizo posible utilizar sólo un sensor para registrar la posición de cada dedo, y obtener a partir del valor registrado por el sensor los ángulos de las articulaciones de los dedos por medio de una tabla de valores construida experimentalmente.

Para la actuación del sistema se utilizaron muelles de Nitinol, cuya velocidad de respuesta y fuerza generada fueron suficientes para movilizar los dedos de un paciente con alto tono muscular. Para conseguir un mejor control sobre los movimientos generados, se hizo uso de 2 muelles por cada dedo, así uno de ellos tiene asignada la extensión del dedo y el otro se hace cargo de la flexión, además de incorporarse un sistema de refrigeración para los actuadores, el cual mejoró considerablemente la velocidad de respuesta de los mismos. A continuación se realiza una tabla comparativa entre las ventajas y desventajas experimentadas al fabricar el sistema con muelles de Nitinol frente a un posible sistema fabricado empleando motores eléctricos:

Tabla 17. Comparación entre el sistema fabricado y un sistema con motores eléctricos.

Sistema con Muelles de Nitinol	Sistema con Motores
Sistema silencioso	Posiblemente se genere ruido debido a los componentes mecánicos que contiene el motor.
Diseño simple que no involucro el uso de componentes mecánicos para acondicionar la extensión y flexión de los dedos.	Diseño de mayor complejidad para su puesta a punto.
Tamaño y peso reducidos (cada muelle pesa 1.1 gr) logrando un peso total de 11 gr en actuadores.	Actualmente en el mercado no se encuentra un motor con un peso menor o igual a 11 gr. Por lo que el sistema fabricado pesara mucho más.
Actuadores Resistente a la corrosión	Con el paso del tiempo los componentes mecánicos comienzan a sufrir oxidación.
Fácil mantenimiento y reparación	Complejidad media para realizar mantenimiento y reparación.
Baja eficiencia energética	Mayor eficiencia energética
Genera altas temperaturas que podrían generar quemaduras.	No contiene componentes que generan calentamiento
Baja velocidad de respuesta	Alta velocidad de respuesta
Control impreciso	Control preciso

Para controlar el paso de energía a los actuadores se implementó una tarjeta de acondicionamiento capaz de administrar las elevadas corrientes que requieren los alambres de Nitinol, y para comunicar al robot con el componente software del sistema, se empleó una tarjeta de adquisición de datos Arduino UNO, cuya interfaz de programación es muy fácil de utilizar.

Para el control del robot, se desarrolló una aplicación software que recibe la lectura realizada sobre los sensores y modifica la cantidad de energía suministrada a los actuadores del sistema. Para esta aplicación se diseñó una interfaz de usuario en la que se puede gestionar archivos con información referente al estado de los pacientes, y programar los ejercicios que se realizan durante las terapias. Además, aprovechando la lectura realizada sobre los sensores, se implementó un *widget* en el cual se muestra un modelo virtual de la mano del paciente que recrea los movimientos desarrollados en la terapia.

Respecto a la aplicación del proyecto se puede concluir que las herramientas de software libre permiten desarrollar aplicaciones complejas y con alto nivel de detalle. La ventaja de utilizar estas herramientas radica en que no es necesario pagar por ellas y no condicionan el uso que se les dé. La característica del software utilizado que mayor valor tuvo para el presente proyecto fue la capacidad de todas las herramientas para integrarse unas con otras, bien fuera por medio de *plugins* o utilizando funciones específicas para ello, lo cual permitió incluir todos los rasgos deseados en la aplicación final.

9. Referencias

- [1] M. Zecca, "On the Development of a Cybernetic Prosthetic Hand," Scuola Superiore Sant' Anna, Pisa, Italia, Tesis de doctorado, 2003.
- [2] L. Zheng, "Using Robotic Hand Technology for the Rehabilitation of Recovering Stroke Patients with Loss of Hand Power," North Carolina State University, Raleigh, Carolina del Norte, Tesis de maestría, 2003.
- [3] C. Schabowsky, "Robot-Assisted Hand Movement Therapy after Stroke," The Catholic University of America, Washington D.C., Tesis de doctorado, 2010.
- [4] J. García Ortiz and D. A. Vallejos Ortiz, "Entorno Gráfico de un Entrenador Virtual de Prótesis de Mano," Universidad del Cauca, Popayán, Colombia, Trabajo de grado, 2012.
- [5] Fernando. "Tendinitis de la extremidad superior - Aparato extensor". [Online]. <http://fernando-espacioamorylocura.blogspot.com/2009/03/tendinitis-de-la-ee-ss-x-mano-iv.html>
- [6] C. Quinayás, "Diseño y construcción de una prótesis robótica de mano funcional adaptada a varios agarres," Universidad del Cauca, Popayán, Colombia, Tesis de Maestría, 2010.
- [7] Family Caregiver Alliance. "Family Caregiver Alliance". [Online]. http://www.caregiver.org/caregiver/jsp/content_node.jsp?nodeid=534
- [8] National Institute of Neurological Disorders and Stroke, "Post-Stroke Rehabilitation," *NIH Publications*, vol. 11, no. 1846, pp. 1-10, abril 2011.
- [9] The Catholic University of America. "Rehabilitation Robotics". [Online]. <http://cabrr.cua.edu/research/RehabilitationRobotics.cfm#HandBot>
- [10] The Neuro Institute. "NeuroInstitute". [Online]. <http://www.theneuroinstitute.com/images/equip/stroke-cva.pdf>
- [11] Kinetic Muscles Inc. "Kinetic Muscles - Improving Life After Stroke". [Online]. <http://www.kineticmuscles.com/hand-physical-therapy-hand-mentor.html>
- [12] Sensory-Motor Systems Lab. "ETH Sensory-Motor Systems Lab - ARMin". [Online]. <http://www.sms.mavt.ethz.ch/research/projects/armin/armin>
- [13] Tyromotion. "Amadeo Hand Rehabilitation". [Online]. <http://www.tyromotion.com/en/products/amadeo.html>
- [14] E. Carletti. Robots. [Online]. http://robots-argentina.com.ar/Actuadores_musculosalambre.htm
- [15] A. Díaz, "Metodología para el desarrollo de dispositivos médicos basados en el empleo de polímeros activos como sensores y actuadores.," Universidad Politécnica de Madrid, España, Tesis Doctoral, Doctor Ingeniero Industrial 2009.
- [16] Imagesco. Imagesco. [Online]. <http://www.imagesco.com/catalog/nitinol/index.html#nities>
- [17] S. Flor, "Simulación numérica y correlación experimental de las propiedades mecánicas en las aleaciones con memoria de forma," Universidad Politécnica de Cataluña, España, Tesis Doctoral, Doctor Ingeniero Industrial 2005.
- [18] P. Linzoain, "Fabricación y caracterización de aleaciones de Ti-Ni con porosidad controlada," Universidad de Sevilla, España, Trabajo de Grado, Ingeniero Industrial 2010.

- [19] G Bizdoaca et al., "On the Design of a Shape Memory Alloy Spring Actuators Using Thermal Analysis," *WSEAS*, vol. 7, no. 10, pp. [1006-1007], Octubre 2008.
- [20] S. Montecinos. (2005) *cabierta*. [Online]. <http://cabierta.uchile.cl/revista/27/articulos/pdf/edu2.pdf>
- [21] Sma-inc. Sma-nc. [Online]. <http://www.sma-inc.com/>
- [22] Frederick Kaiser. [Online]. <http://fkaiser.pbworks.com/w/page/27565854/Paper%202>
- [23] K. Otsuka, "An introduction to the R phase transition.," in *Engineering aspects of shape memory alloys*. Londres: Butterworth-Heinemann.
- [24] Brent. y McAvoy, J. Taft. (2006) Determination of the temperature Dependent Spring Constant of a Nitinol Expansion Spring. [Online]. <http://www.imagesco.com/science/Nitinol-Spring%20Data.pdf>.
- [25] A. Cano, "Estudio e implementacion de actuadores basados en aleaciones SMA," Universidad Carlos III, España, Trabajo de grado, Ingeniero Industrial 2010.
- [26] G. Ramírez, "Estudio Comparativo de Actuadores SMA vs Motor DC. Diseño y Construcción de una Plataforma de Pruebas," Universidad Carlos II, España, Trabajo de Grado, Ingeniero Industrial 2012.
- [27] MuscleWires. MuscleWires. [Online]. <http://www.musclewires.com/Products.php>
- [28] Dynalloy, Inc. Dynalloy. [Online]. <http://www.dynalloy.com/TechDataWire.php>
- [29] OpenSurg. "Opensurg". [Online]. <http://isa.umh.es/opensurg/>
- [30] Blender Foundation. "Blender". [Online]. <http://www.blender.org/>
- [31] Autodesk. "Productos Autodesk Inventor". [Online]. <http://www.autodesk.es/adsk/servlet/pc/index?id=14569016&siteID=455755>
- [32] Ogre 3D Team. "Ogre Wiki - Support and Documentation for Ogre 3D". [Online]. <http://www.ogre3d.org/tikiwiki/>
- [33] Comunidad de Ogre en español. "OgreES Wiki". [Online]. <http://ogrees.wikispaces.com/>
- [34] Celularis. "Celularis". [Online]. <http://www.celularis.com/software/13-razones-para-utilizar-qt/>
- [35] Wikipedia. "Wikipedia". [Online]. <http://es.wikipedia.org/wiki/Wikipedia:Portada>
- [36] C. Muñoz, M. Vivas, O. Gaviria, C. Quinayás, "Diseño y construcción de la prótesis robótica de mano UC-1," *Ing. Univ. Bogotá*, vol. 14, no. 2, pp. 223-237, noviembre 2010.
- [37] 3D Systems. "3D Systems - ProJet HD3000 and HD3000plus Professional 3D Printer". [Online]. <http://printin3d.com/3d-printers/projet-hd-3000-3d-professional-3d-printer>
- [38] Igus. "Igus - Plastics for longer life". [Online]. <http://www.igus.es/default.asp?Page=6>
- [39] Asapar. "Asapar". [Online]. <http://www.siliconpc.com/asapar/agua.htm>
- [40] 3D Systems. "3D Systems - ProJet 3000 and 5000 Printer Materials". [Online]. <http://printin3d.com/3d-printer-materials/projet-3000-5000-3d-printer-materials>
- [41] Arduino. Arduino. [Online]. <http://www.arduino.cc/es/>
- [42] Mouser Electronics. Mouser. [Online]. <http://www.mouser.com/ProductDetail/Avago-Technologies/HEDS-9720L52/?qs=pQfy5%252bKCabJ%252bibQtMvtdNg%3d%3d>
- [43] TEM Electronic Components. [Online]. http://www.tme.eu/es/katalog/potenciometros-deslizantes_112333/
- [44] Cadsoft. [Online]. <http://www.cadsoftusa.com/>
- [45] Vishay. [Online]. <http://www.vishay.com/docs/91019/91019.pdf>

- [46] Nokia. "Qt". [Online]. <http://qt.nokia.com/qt-in-use/story/app/kde>
- [47] T. Roosendaal and S. Selleri. "Manual de Blender en Español". [Online]. http://www.futureworkss.com/tecnologicos/informatica/tutoriales/Manual_de_Blender.pdf
- [48] M. Muldoon. "Blend Swap". [Online]. <http://www.blendswap.com/>
- [49] M. Reimpell and L. Pang. "Ogre Wiki: Suport and community documentation for Ogre 3D". [Online]. <http://www.ogre3d.org/tikiwiki/Blender+Exporter>
- [50] Ogre 3D Team. "Ogre - API Reference". [Online]. <http://www.ogre3d.org/docs/api/html/index.html>
- [51] Comunidad pugixml. "pugixml". [Online]. <http://pugixml.org/>
- [52] Nelson Sotomayor. [Online]. http://es.scribd.com/erodriguez_139/d/55968566/31-CONTROL-ON-OFF