UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y REDES DE COMUNICACIÓN

"DETECCIÓN DE ÁNGULO EN MOVIMIENTOS CERVICALES PARA LA REHABILITACIÓN DEL ADULTO MAYOR MEDIANTE VISIÓN ARTIFICIAL"

TRABAJO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y REDES DE COMUNICACIÓN

AUTOR: ALFREDO STALIN IBARRA GER

DIRECTOR: MSC. EDGAR ALBERTO MAYA OLALLA

IBARRA-ECUADOR

2025

UNIVERSIDAD TÉCNICA DEL NORTE

BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO			
CÉDULA DE IDENTIDAD:	0401485321		
APELLIDOS Y NOMBRES:	Ibarra Ger Alfredo Stalin		
DIRECCIÓN:	Tulcán – Ciudadela Pullman Carchi		
EMAIL:	asibarrag@utn.edu.ec / stalinibarrag@outlook.com		
TELÉFONO FIJO:		**TELÉFONO MÓVIL:**	0994495026

DATOS DE LA OBRA	
TÍTULO:	DETECCIÓN DE ÁNGULO EN MOVIMIENTOS CERVICALES PARA LA REHABILITACIÓN DEL ADULTO MAYOR MEDIANTE VISIÓN ARTIFICIAL
AUTOR (ES):	Ibarra Ger Alfredo Stalin
FECHA: DD/MM/AAAA	26/02/2025
SOLO PARA TRABAJOS DE GRADO	
PROGRAMA:	■ **PREGRADO** ☐ **POSGRADO**
TITULO POR EL QUE OPTA:	Ingeniero en Electrónica y Redes de Comunicación
DIRECTOR:	MSC. Edgar Alberto Maya Olalla
ASESOR:	MSC. Ana Cristina Umaquinga Criollo

2. CONSTANCIAS

El autor manifiesta que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto, la obra es original y que es el titular de los derechos patrimoniales, por lo que asume la responsabilidad sobre el contenido de la misma y saldrá en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 26 días del mes de febrero de 2025.

EL AUTOR:

Alfredo Stalin Ibarra Ger

CI: 0401485321

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CERTIFICACIÓN:

MAGÍSTER EDGAR MAYA, DIRECTOR DEL PRESENTE TRABAJO DE TITULACIÓN CERTIFICA:

Que el presente trabajo de Titulación DETECCIÓN DE ÁNGULO EN MOVIMIENTOS CERVICALES PARA LA REHABILITACIÓN DEL ADULTO MAYOR MEDIANTE VISIÓN ARTIFICIAL, ha sido desarrollado por el señor Ibarra Ger Alfredo Stalin bajo mi supervisión.

Es todo cuanto puedo certificar en honor a la verdad.

Msc. Edgar Alberto Maya Olalla

DIRECTOR

DEDICATORIA

Dedico este trabajo con todo mi amor a mis padres Alfredo Ibarra y Pilar Ger, quienes con su esfuerzo, apoyo incondicional y sus enseñanzas han sido mi mayor inspiración.

A mis apreciadas hermanas Evelyn y Carolina, por su apoyo y compañía en cada paso de este camino.

A la memoria de mi querida abuelita, cuyo amor y enseñanzas me han acompañado siempre a lo largo de mi vida, tu legado vive en cada uno de mis logros.

Sin mi familia no hubiera logrado este gran éxito.

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

AGRADECIMIENTO

Le agradezco a mi querida enamorada, Naomi Portilla, por su apoyo incondicional a lo largo de este proceso. Su paciencia, comprensión y aliento constante fueron fundamentales en los momentos de mayor desafío.

A mis amigos, Max A., Max C., Kevin E., Kevin O., Cristian, Ángel, Mila, July, Mishel y Caro, por su apoyo inquebrantable y por hacer este viaje académico más llevadero, no cabe duda de que fueron los mejores compañeros en esta etapa de mi vida.

A mis profesores por compartir su conocimiento y guía, que fueron clave en mi formación como profesional. En especial a mi tutor MSC. Edgar Maya y mi codirectora MSC. Ana Umaquinga, por su paciencia y confianza en mí.

RESUMEN

Los trastornos de movilidad cervical en el adulto mayor representan un desafío en la sociedad moderna y una carga económica significativa. La rehabilitación a través de fisioterapia emplea diversas técnicas para mejorar la movilidad del paciente, enfocándose en la recuperación del rango de movimiento cervical mediante sesiones especializadas. La evaluación de los ángulos articulares en los movimientos cervicales se realiza tradicionalmente de forma manual, utilizando herramientas como graduadores o goniómetros, las cuales pueden generar cierto margen de error debido a la estimación visual del evaluador. Por ello, la incorporación de tecnología en la medición de estos ángulos cobra relevancia, permitiendo una evaluación más precisa y objetiva del progreso en la rehabilitación.

Este proyecto tiene como objetivo diseñar e implementar una aplicación capaz de estimar la posición de las extremidades en 2D, permitiendo así calcular los ángulos de los distintos movimientos cervicales en pacientes adultos mayores. Para ello, se empleará tecnología de Visión Artificial, la cual facilitará la extracción de datos en tiempo real a partir de imágenes capturadas por una cámara de alta definición. Esta cámara estará conectada a un ordenador encargado de procesar el video del paciente. Una vez obtenidos los datos de postura mediante la detección de puntos clave, estos se parametrizarán como vectores, lo que permitirá calcular con precisión los ángulos cervicales requeridos.

Los datos digitales obtenidos a través de la goniometría del raquis cervical permiten establecer el punto de partida del tratamiento, evaluar su progreso a lo largo del tiempo y determinar su finalización una vez que el paciente haya logrado una recuperación completa.

INDICE DE CONTENIDOS

INDICE DE FIGURAS

INDICE DE TABLAS

1. Antecedentes

En el capítulo de antecedentes se especifica los requerimientos necesarios para la elaboración del presente trabajo de titulación, tales como: el tema abordado, el problema a solventar, los objetivos planteados, el alcance y la justificación. Esto con el fin de sustentar la investigación realizada para el proyecto.

1.1.Tema

DETECCIÓN DE ÁNGULO EN MOVIMIENTOS CERVICALES PARA LA REHABILITACIÓN DEL ADULTO MAYOR MEDIANTE VISIÓN ARTIFICIAL.

1.2.Problema

Los adultos mayores presentan alteraciones osteomusculares en su movilidad cervical propias de la edad, debido a patologías o causadas por la inactividad física, produciendo rigidez articular y disminución de la fuerza muscular dificultando sus actividades cotidianas (Taboadela C., 2016). En el Ecuador, los adultos mayores son considerados como una población no productiva y vulnerable en muchos aspectos, a su edad hay poca absorción de proteínas por lo que en sus músculos se reporta pérdida de masa muscular, así mismo atrofia muscular por pérdida gradual de fibras musculares asociada a disminución de la fuerza entre los 50 y 70 años, por lo tanto hay una disminución de la capacidad funcional producido por la debilidad que limita sus desplazamientos y realización de sus actividades de la vida diaria. “Los trastornos de movilidad cervical son un problema en la sociedad moderna además de ser una carga económica importante, debido a los costos de la licencia por enfermedad y atención de la salud; estos afectan a personas en todo el mundo” (Arévalo, 2014).

Por otro lado, La palabra goniometría deriva del griego gonion (‘ángulo’) y metron (‘medición’), es decir: «disciplina que se encarga de estudiar la medición de los

ángulos». La goniometría ha sido utilizada por la civilización humana desde la antigüedad hasta nuestro tiempo en innumerables aplicaciones, como la agricultura, la carpintería, la herrería, la matemática, la geometría, la física, la ingeniería, la arquitectura, la medicina, entre otras (Taboadela H. , 2007).

La rehabilitación a través de fisioterapia es realizada por técnicas que permiten la recuperación del paciente por medio de sesiones en las cuales se busca mejorar el ángulo de movilidad cervical. La medición de ángulos articulares en movimientos cervicales es evaluado manualmente, con herramientas como lo son graduadores o goniómetros que proporcionan cierto error de medición por estimación visual propia del ser humano, es así que toma importancia la inclusión de la tecnología para determinar el avance en la rehabilitación de movilidad cervical (Taboadela H., 2007). El profesional encargado de la rehabilitación física del paciente realiza mediciones continuas en cada sesión para determinar el punto de inicio del tratamiento, dependiendo de la patología, así como también determinar las secuelas y el punto final del tratamiento. Mediante este proyecto, se quiere implementar un nuevo método de rehabilitación o más bien poco usado en la fisioterapia en el Ecuador. Según Leack (2016) señalo que “la mejoría en los ángulos de movilidad cervical ha demostrado ser un gran ejercicio capaz de proporcionar una terapia para las personas que sufren de dolor en las articulaciones y la espalda”.

La solución planteada a esta problemática está basada en la aplicación de un modelo de algoritmo pre - entrenado de visión artificial que permite la determinación de posición estimada de extremidades en 2D, lo cual permite calcular el ángulo de movilidad cervical en el paciente adulto mayor. La recuperación de los pacientes es determinada por diversos factores entre los que se cita: la constancia y los métodos aplicados por el fisioterapeuta tratante, la integración de la tecnología en este tipo de actividades permite la optimización de recursos y una mejor planificación gracias al uso de la inteligencia

artificial (Taboadela H. , 2007). Esto permite la interacción del paciente con un ambiente tecnológico con el objetivo de recuperar la agilidad que se pierde en el proceso de envejecimiento y optimizar su calidad de vida.

1.3.Objetivos

1.3.1. Objetivo General

Determinar los diferentes ángulos de movilidad cervical en el plano sagital, frontal y vertical para la rehabilitación del adulto mayor utilizando visión artificial y mostrar los resultados en una interfaz de usuario para compararlos con los medidos manualmente.

1.3.2. Objetivos Específicos

- Analizar bibliografía adecuada respecto a las diferentes maneras de evaluar la movilidad cervical en el área de la fisioterapia.
- Diseñar e implementar un sistema que permita evaluar el ángulo de movilidad cervical en el adulto mayor utilizando un algoritmo de estimación de posición mediante inteligencia artificial.
- Diseñar e implementar una interfaz de usuario que permita visualizar los resultados obtenidos al profesional tratante con el fin de que el mismo pueda seleccionar las mejores actividades de rehabilitación para el paciente.
- Comprobar que los resultados obtenidos sean confiables en comparación con los medidos de manera manual mediante un goniómetro, demostrando así que el sistema diseñado sea viable.

1.4.Alcance

El sistema planteado se basa en la medición de ángulos cervicales mediante un algoritmo pre-entrenado que permita la estimación de poses, dicho algoritmo manipula articulaciones específicas dentro del cuerpo humano. Estas articulaciones se conocen como “puntos clave” dentro del sistema de estimación de pose, los puntos clave de estimación de pose generalmente se alinean con articulaciones humanas reales como el codo, la muñeca, la rodilla, entre otros. Y a través de estos puntos es posible realizar la parametrización de los ángulos y la posición de las articulaciones como vectores (Jarrín Chacón, 2020).

Para esto, en primer lugar, se realizará un análisis literario con el fin de comprender la goniometría o medición de ángulos del sistema osteoarticular para aplicarlo a la valoración de las incapacidades laborales en el adulto mayor. Una vez comprendido los movimientos del cuerpo humano que se desea evaluar se posibilita el diseño y la implementación de un sistema que pueda medir el ángulo de movilidad cervical en el paciente.

El desarrollo del proyecto se realizará a través del área de Visión Artificial que permitirá la extracción de datos que estimen poses en tiempo real de imágenes obtenidas mediante el uso de una cámara de alta definición. Dicha cámara se deberá comunicar con el ordenador que a su vez deberá procesar el video capturado del paciente, una vez obtenidos los datos de postura (puntos clave), se parametrizarán como vectores que permitirán el cálculo de los ángulos cervicales que se buscan obtener.

A continuación, mediante una interfaz de usuario se indicará un esquema donde se pueda visualizar los grados de movimiento cervical en el paciente y a su vez dichos ángulos se guardarán en una base de datos con la debida descripción del paciente.

Para finalizar se realizará las diferentes pruebas a un determinado número de pacientes del Centro de Gerontológico Residencial León Ruales, los datos obtenidos se deben comparar con los medidos de manera manual mediante el uso de un goniómetro y de ser necesario se tendrá que corregir los errores del sistema con el fin de garantizar que las mediciones realizadas brinden datos acertados y la implementación del sistema sea factible.

1.5.Justificación

La fisioterapia es el tratamiento de enfermedades, lesiones o deformidades mediante métodos físicos como masajes, tratamiento térmico y ejercicio, en lugar de medicamentos o cirugía. Envejecer hace que nuestro cuerpo pase por muchos cambios físicos. Estos cambios suelen provocar una disminución de la fuerza muscular, la densidad ósea, la coordinación corporal e incluso hacer que las articulaciones se vuelvan más rígidas, lo que en ocasiones puede provocar caídas y fracturas (Muñoz Ruiz, 2017). Para las personas mayores, el ejercicio puede ser la clave para recuperar y mantener la función física requerida en la vida diaria. Los programas de ejercicio establecidos por un fisioterapeuta pueden ayudar a reducir el dolor corporal, mejorar el movimiento de las articulaciones, facilitar la coordinación y estimular la función respiratoria. Aunque la fisioterapia no puede detener el proceso de envejecimiento, puede ayudar a reducir el impacto que tiene en nuestros cuerpos (Leack, 2016).

El envejecimiento de la población mundial es un fenómeno que marcará el siglo XXI. A escala global, cada segundo 2 personas cumplen 60 años y al momento existen 810 millones de personas en el mundo mayores de esa edad (Kolber M., 2013). En el Ecuador existen 1.049.824 personas mayores de 65 años (6,5% de la población total). La distribución por género de la población nacional adulta mayor es de 53% para las mujeres

y de 47% para los hombres (INEC, 2021). El Reglamento General de la Ley Orgánica de las personas Adultas Mayores tiene como objeto establecer los lineamientos, directrices y normas para la aplicación de la Ley Orgánica de las Personas Adultas Mayores y para el funcionamiento, control y seguimiento del Sistema Nacional Especializado de Protección Integral de los Derechos de las Personas Adultas Mayores, así como establecer los mecanismos para la prevención, atención, protección, restitución y reparación a las personas adultas mayores (Reglamento General Ley Orgánica de las personas Adultas Mayores, 2020).

Gracias a la tecnología moderna, la ciencia médica avanza a un ritmo sin precedentes en ningún momento de la historia. Prácticamente todos los días surgen nuevas tecnologías en todo el mundo, que abren nuevos mundos tanto para los pacientes como para los profesionales médicos. A medida que aumenta el volumen de pacientes de rehabilitación en todo el espectro de edad, los médicos se ven obligados a producir resultados favorables con recursos y tiempo limitados, razón por la cual se ha producido una proliferación de nuevas tecnologías en rehabilitación con la esperanza de mejores resultados. Todos los campos especializados de la ciencia médica han logrado grandes avances en los últimos años. Algunas que se utilizan en la fisioterapia hoy en día son: Performance Matrix, escáneres de Marcha, análisis de carrera 3D, la electromiografía, la realidad virtual, entre otros (Muñoz Ruiz, 2017).

2. Fundamentación Teórica

2.1.Goniometría

El arte y la ciencia de medir los rangos articulares en cada plano de la articulación se denominan goniometría. El término 'goniometría' tiene su origen en dos palabras griegas, ‘gonia’, que significa ángulo, y ‘metron’, que significa medir. (Gandbhir & Cunha, 2022). La goniometría ha sido utilizada por los humanos desde la antigüedad hasta nuestros días en innumerables aplicaciones, tales como: la agricultura, la carpintería, las matemáticas, la geometría, la física, la ingeniería, la arquitectura, el estudio de la medicina, entre otras (Taboadela, 2007).

Milanese et al. (2014) afirma que la goniometría es una habilidad de evaluación esencial en la práctica musculoesquelética, con las medidas resultantes utilizadas para determinar la presencia o ausencia de disfunción, guiar intervenciones de tratamiento y generar evidencia de la efectividad del tratamiento en el paciente.

Se trata de una técnica sencilla, accesible, no invasiva y de bajo costo, comúnmente empleada por cirujanos ortopédicos y fisioterapeutas para valorar la magnitud de las lesiones articulares y monitorear la progresión clínica del paciente. (Reusing et al., 2020). En la rehabilitación del adulto mayor, la goniometría se utiliza para determinar el punto de partida del tratamiento, evaluar el curso del tratamiento a lo largo del tiempo, motivar al paciente, determinar el pronóstico, modificar o finalizar el tratamiento y, finalmente, evaluar los efectos (Taboadela, 2007).

2.2.Instrumentos De Medición

Para medir el rango de movimiento, los médicos, quiroprácticos, fisioterapeutas u otros profesionales de la salud suelen utilizar diferentes dispositivos capaces de medir el movimiento angular de una articulación tales como:

2.2.1. Goniómetro

Un goniómetro es un dispositivo que mide un ángulo o permite la rotación de un objeto a una posición definida. Dos Santos et al., (2017) explica que al ser una herramienta no invasiva, económica y sobre todo de uso sencillo, el goniómetro es un método que mide los rangos de movimientos, con una buena relación entre costo y beneficio.

Un goniómetro universal tiene tres partes, tal como se puede ver en la Figura 1:

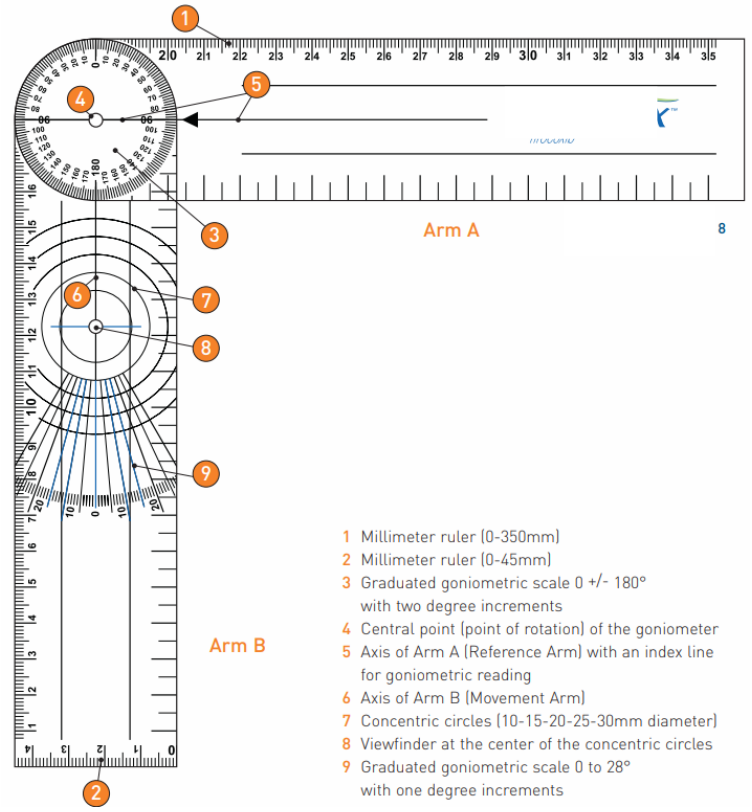
- Un **cuerpo:** el cual está diseñado como un transportador (graduador) y puede formar un semicírculo o un círculo completo. Tiene una escala para la medida del ángulo. La escala puede extenderse de 0 a 180 o de 180 a 0 grados sexagesimales para modelos de medio círculo o de 0 a 360 grados sexagesimales en modelos de círculo completo (Gandbhir & Cunha, 2022).
- El **punto de apoyo:** es un tornillo en el centro del cuerpo que permite que el brazo móvil se mueva libremente en el cuerpo del dispositivo. El dispositivo similar a un tornillo puede apretarse para fijar el brazo móvil en una posición particular o aflojarse para permitir el libre movimiento (Reusing et al., 2020).
- El **brazo estacionario:** es el brazo del goniómetro que se alinea con la parte inactiva de la articulación medida. El brazo móvil es el brazo del

goniómetro, que se alinea con la parte móvil de la articulación medida (Petazzoni & Jaeger, 2008).

Un goniómetro puede evaluar tanto el rango de movimiento activo como el pasivo. El posicionamiento juega un papel vital en la goniometría porque ayuda a colocar las articulaciones en una posición inicial cero o en una posición neutral y ayuda a estabilizar el segmento articular proximal. El examinador estabiliza el componente articular proximal y luego mueve con cuidado el componente distal de la articulación a lo largo de todo su rango de movimiento disponible hasta alcanzar la sensación final (Gandbhir & Cunha, 2022).

Figura 1

Goniómetro y sus partes



Fuente: (Petazzoni & Jaeger, 2008)

2.2.2. Electro Goniómetro

Un electro goniómetro es un dispositivo eléctrico que se utiliza para evaluar la flexibilidad y la movilidad de una articulación. Los electro goniómetros son muy utilizados en centros de rehabilitación y tambien de investigación. Las mediciones obtenidas de este instrumento son fiables ya que permiten controlar los distintos rangos de movimiento de las articulaciones y posteriormente determinar las alteraciones motoras de los pacientes. (Grecheneva, A. et al., 2019)

Sigue posiciones angulares a través de planos distintivos de movimiento a través de accesorios tópicos ubicados en varias articulaciones del cuerpo para monitorear la conversión de señales de voltaje, en respuesta a movimientos dinámicos. En la Figura 2, se puede apreciar que sus mediciones se registran en un ordenador a través de un determinado software mismo del instrumento (McKinnon et al., 2021).

Figura 2

Electro goniómetro

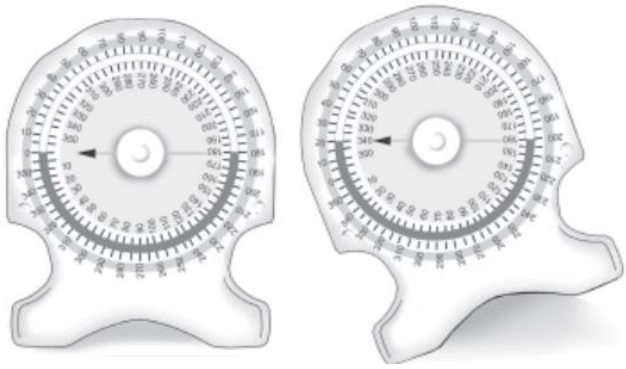
Fuente: (Taboadela, 2007)

2.2.3. Inclinómetro

El inclinómetro (ver Figura 3) es un dispositivo utilizado para medir ángulos cuando no es posible utilizar el goniómetro, por ejemplo, cuando se mide la flexión y extensión lumbar, o cuando es difícil determinar la remodelación ósea. El inclinómetro utiliza la gravedad como referencia para la calibración. Como resultado, la posición inicial de la medición no depende del juicio visual, como ocurre con un goniómetro, y puede repetirse sin problemas (Tozzo et al., 2021).

Figura 3

Inclinómetro



Fuente: (Taboadela, 2007)

2.3.Planimetría

Dentro de la fisioterapia se utiliza la planimetría como una manera de describir los planos sobre los cuales se producen los movimientos de las diferentes articulaciones. Para facilitar el estudio de estos movimientos en el cuerpo humano se contemplan tres planos cruzados de manera perpendicular entre sí: plano sagital, plano frontal y plano transversal (Petazzoni & Jaeger, 2008).

2.3.1. Plano Sagital

El plano sagital divide al cuerpo humano en dos partes, una a la derecha y otra a la izquierda, tal como muestra la Figura 4. En caso de existir más planos paralelos a éste se denominan de la misma forma o a su vez parasagitales (Pasqual, 2003).

2.3.2. Plano Frontal

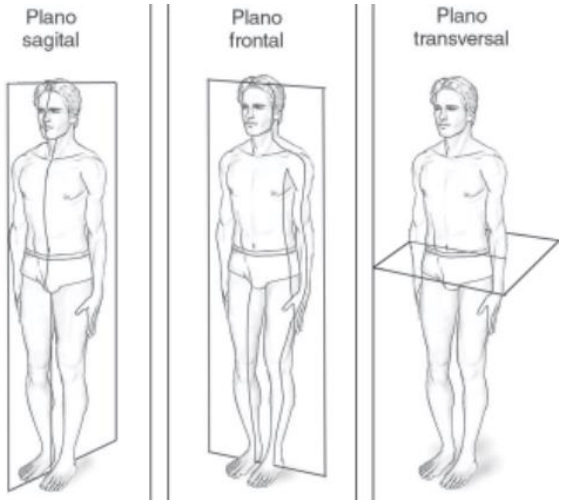
El plano frontal o también conocido como plano coronal es aquel que se encuentra perpendicular al plano sagital y de igual manera divide al cuerpo humano en dos partes, una anterior y otra posterior como se muestra en la Figura 4. Se puede observar los movimientos realizados en este plano desde el frente del paciente (Taboadela, 2007).

2.3.3. Plano Transversal

El plano transversal u horizontal es aquel plano que se encuentra perpendicular a los otros 2 planos (sagital y frontal) y se encarga de dividir el cuerpo en dos partes, una superior y una inferior como se puede observar en la Figura 4. Los movimientos realizados en este plano pueden visualizarse desde la parte de arriba o abajo del paciente (Petazzoni & Jaeger, 2008).

Figura 4

Planos de movimiento en el cuerpo humano



Fuente: (Taboadela, 2007)

2.4.Goniometría Del Raquis Cervical

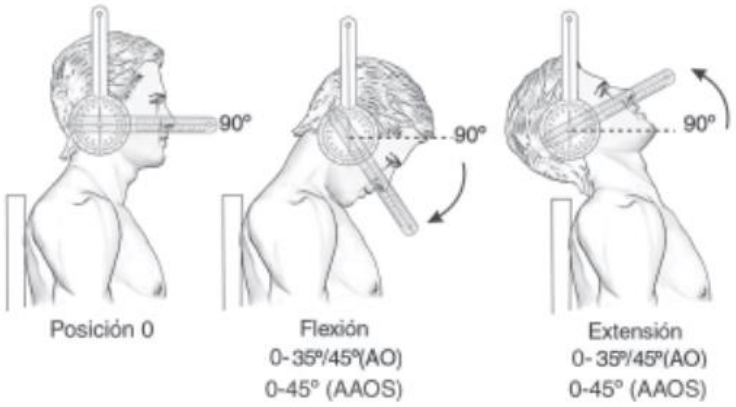
Según Taboadela (2007): “La columna cervical se extiende desde la articulación occipitoatloidea hasta la articulación entre la séptima vértebra cervical y la primera vértebra torácica”. Los movimientos de la columna cervical se realizan en todos los planos antes mencionados, en el plano sagital (flexión y extensión), en el plano frontal (inclinación lateral derecha e izquierda) y en el plano vertical (rotación derecha e izquierda) (Pasqual, 2003).

2.4.1. Flexión-Extensión

Para realizar este movimiento, en primer lugar, se debe colocar al paciente sentado con los pies totalmente rectos en el piso y con la espalda apoyada en el respaldo de la silla, lo que puede ser considerada como posición 0. Desde la posición 0 como indica la Figura 5, se realiza un movimiento hacia adelante (flexión) y luego hacia atrás (extensión). Los ángulos que se deben registrar son aquellos que se forman desde la posición 0 y las posiciones finales de flexión y extensión del paciente, además estos datos deben ser vistos desde el plano sagital (Araujo et al., 2024).

Figura 5

Movimientos de flexión – extensión de la columna cervical



Fuente: (Taboadela, 2007)

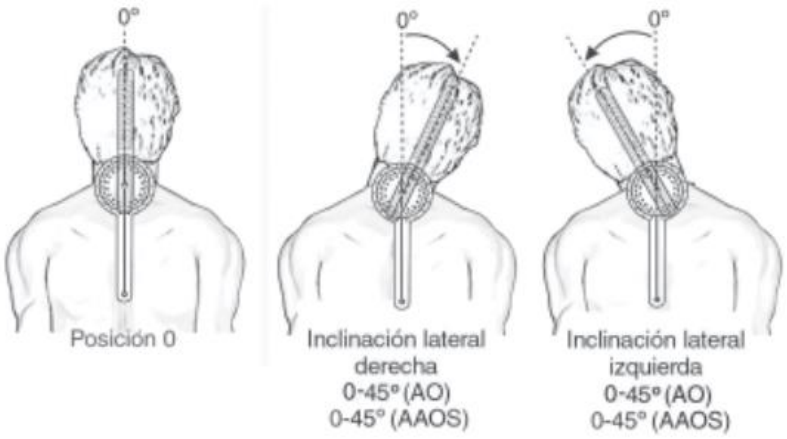
2.4.2. Inclinación Lateral Derecha e Izquierda

Para realizar este movimiento, en primer lugar, se debe colocar al paciente sentado con la espalda apoyada en el respaldo de la silla y mirando al frente. Desde la posición 0 como indica la Figura 6, se realiza un movimiento lateral hacia la izquierda y luego hacia la derecha cuidando que el paciente no mueva los hombros ni incline la cabeza.

Los ángulos que se deben registrar son aquellos que se forman desde la posición 0 y las posiciones finales de inclinación lateral hacia la izquierda y hacia la derecha del paciente, además estos datos deben ser vistos desde la parte posterior del plano frontal (Sukari et al., 2021).

Figura 6

Movimientos de inclinación lateral derecha e izquierda



Fuente: (Taboadela, 2007)

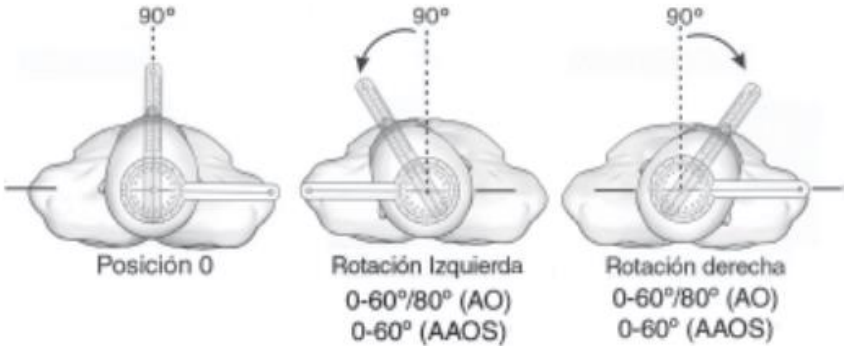
2.4.3. Rotación Derecha e Izquierda

Para realizar este movimiento, en primer lugar, se debe colocar al paciente sentado con la espalda apoyada en el respaldo de la silla y mirando al frente. Desde la posición 0 como indica la Figura 7, se realiza un movimiento de rotación tanto a la derecha como a la izquierda cuidando que el paciente no mueva los hombros.

Los ángulos que se deben registrar son aquellos que se forman desde la posición 0 y las posiciones finales de rotación hacia la derecha y hacia la izquierda del paciente, además estos datos deben ser vistos desde la parte superior del plano transversal (Youdas et al., 2011).

Figura 7

Movimientos de rotación derecha e izquierda de la columna cervical



Fuente: (Taboadela, 2007)

2.5.Inteligencia Artificial

Según Mccarthy, (2007): Dentro de ingeniería la inteligencia artificial es una ciencia encargada de fabricar máquinas inteligentes, especialmente programas informáticos inteligentes. Esto plantea el mismo desafío en el uso de computadoras para comprender la inteligencia humana, pero la Inteligencia Artificial (IA) no debe limitarse a métodos que son biológicamente observables.

Sin embargo, Russell & Norvig, (2022) luego procedieron a publicar Inteligencia Artificial: un enfoque moderno, donde profundizan en cuatro posibles objetivos o definiciones de la IA, que diferencia los sistemas informáticos sobre la base de la racionalidad y el pensamiento frente a la actuación: **Enfoque humano** (Sistemas que

piensan como humanos y Sistemas que actúan como humanos), **Enfoque ideal** (Sistemas que piensan racionalmente y Sistemas que actúan racionalmente).

En su forma más simple, la inteligencia artificial es un campo que combina la informática y conjuntos de datos sólidos para permitir la resolución de problemas. También abarca subcampos de machine learning y deep learning, que se mencionan con frecuencia junto con la inteligencia artificial. Estas disciplinas están compuestas por algoritmos de IA que buscan crear sistemas expertos que hagan predicciones o clasificaciones basadas en datos de entrada (IBM, 2012a).

2.5.1. Aprendizaje Automático (Machine Learning)

El aprendizaje automático es un campo de la inteligencia artificial y la informática que se basa en el uso de datos y algoritmos para replicar el proceso de aprendizaje humano, optimizando progresivamente su precisión. A través de técnicas estadísticas, los algoritmos se entrenan para realizar predicciones o clasificaciones, así como para identificar información relevante en proyectos de datos. Estos conocimientos luego contribuyen a la toma de decisiones en diversas aplicaciones, con el objetivo de influir positivamente en métricas clave de crecimiento. (IBM, 2012c).

El aprendizaje automático utiliza modelos algorítmicos que permiten que una computadora aprenda a sí misma sobre el contexto de los datos visuales. Si se alimentan suficientes datos a través del modelo, la computadora "mirará" los datos y aprenderá a diferenciar una imagen de otra. Los algoritmos permiten que la máquina aprenda por sí misma, en lugar de que alguien la programe para reconocer una imagen (Smola & Vishwanathan, 2008).

El aprendizaje automático enfocado al análisis y clasificación de datos médicos ayuda a sistematizar dosis de medicamentos, procesos de rehabilitación, prevención de patologías en pacientes, entre otras actividades. Los modelos de aprendizaje automático son algoritmos que cumplen con tareas específicas y se dividen en 3 grupos principales, **train** (ingreso de datos y asignación de tarea) **validación** (evaluación de los resultados de la tarea) **prueba** (en base a los resultados obtenidos se puede determinar el comportamiento del algoritmo con nuevos datos a futuro) (Pineda, 2022).

2.5.1.1. Aprendizaje Profundo (Deep Learning): El aprendizaje profundo es un subconjunto del aprendizaje automático, que es esencialmente una red neuronal con tres o más capas. Estas redes neuronales intentan simular el comportamiento del cerebro humano, aunque lejos de igualar su capacidad, lo que le permite "aprender" de grandes cantidades de datos. Si bien una red neuronal con una sola capa aún puede hacer predicciones aproximadas, las capas ocultas adicionales pueden ayudar a optimizar y refinar la precisión (Sarker, 2021).

El aprendizaje profundo impulsa muchas aplicaciones y servicios de inteligencia artificial que mejoran la automatización, realizando tareas analíticas y físicas sin intervención humana. En el aprendizaje profundo las redes neuronales (algoritmos inspirados en el cerebro humano) aprenden de grandes cantidades de datos. Los algoritmos de aprendizaje profundo realizan una tarea repetidamente y mejoran gradualmente el resultado a través de capas profundas que permiten el aprendizaje progresivo (MATLAB, 2015).

Dichas capas no son creadas o diseñadas por humanos, sino, por el mismo algoritmo. En la primera capa, generalmente se agrupa las características más importantes

de los datos. En la segunda capa, se agrupan las características de la capa anterior junto los detalles más relevantes. Desde la tercera capa en adelante, el algoritmo es capaz de reconocer y ensamblar combinaciones basadas en las capas anteriores. Este tipos de modelos son mucho más rápidos que otros modelos, ya que pueden modificarse a si mismos para cumplir con una tarea asignada (Lecun et al., 2015).

2.5.1.2.Redes Neuronales.

Las redes neuronales, también conocidas como redes neuronales artificiales traducido del inglés Artificial Neural Networks (ANN) o redes neuronales simuladas del inglés Simulated Neural Networks (SNN), son un subconjunto del aprendizaje automático y están en el corazón de los algoritmos de aprendizaje profundo (Li et al., 2021). Su nombre y estructura están inspirados en el cerebro humano, imitando la forma en que las neuronas biológicas se envían señales entre sí. Una red neuronal es solo una colección de unidades conectadas entre sí; las propiedades de la red están determinadas por su topología y las propiedades de las "neuronas" (Castaneda Sanchez et al., 2023) .

Las redes neuronales artificiales (ANN) se componen de capas de nodos, que contienen una capa de entrada, una o más capas ocultas y una capa de salida. Cada nodo, o neurona artificial, se conecta con otro y tiene asociado un peso y un umbral. Si la salida de cualquier nodo individual está por encima del valor de umbral especificado, ese nodo se activa y envía datos a la siguiente capa de la red. De lo contrario, no se pasan datos a la siguiente capa de la red (Basheer & Hajmeer, 2000).

Las redes neuronales se basan en datos de entrenamiento para aprender y mejorar su precisión con el tiempo (Olabe, 2008). Sin embargo, una vez que estos algoritmos de aprendizaje se ajustan con precisión, se convierten en herramientas poderosas en

informática e inteligencia artificial, lo que nos permite clasificar y agrupar datos a alta velocidad (Li et al., 2021).

2.5.1.3.Red Neuronal Convolucional (CNN).

Una red neuronal convolucional traducido del ingles Convolutional Neural Networks (CNN), está constituida por varias redes neuronales artificiales. Actualmente se puede determinar que este tipo de redes facilitan el procesamiento, clasificación y análisis de imágenes similar a como el cerebro humano lo hace, identificando aspectos como color, bordes, curvas, sombras (Lubinus Badillo et al., 2021).

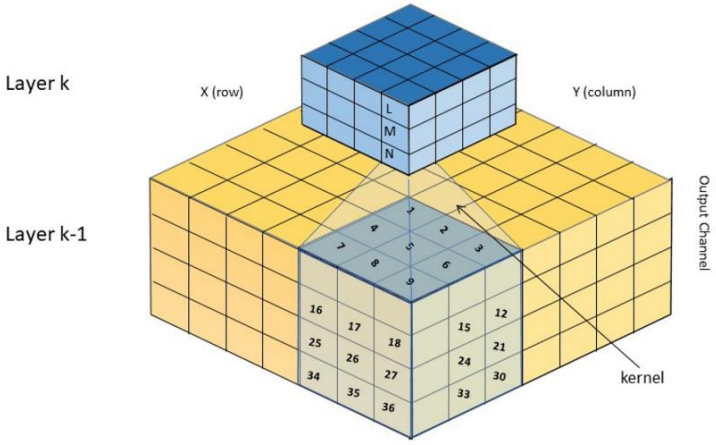
Las imágenes están formadas por píxeles. Cada píxel está representado por un número entre 0 y 255. Por lo tanto, cada imagen tiene una representación digital que es cómo las computadoras pueden trabajar con imágenes. Una CNN ayuda a un modelo de aprendizaje automático o de aprendizaje profundo a "mirar" al dividir las imágenes en píxeles a los que se asignan etiquetas (Chollet, 2017). Utiliza las etiquetas para realizar circunvoluciones (una operación matemática en dos funciones para producir una tercera función) y hace predicciones sobre lo que está "viendo". La red neuronal ejecuta circunvoluciones y verifica la precisión de sus predicciones en una serie de iteraciones hasta que las predicciones comienzan a hacerse realidad. Es entonces reconocer o ver imágenes de una manera similar a los humanos (Kim & Reiter, 2017).

Al igual que un ser humano que distingue una imagen a distancia, una CNN primero discierne bordes duros y formas simples, luego completa la información a medida que ejecuta iteraciones de sus predicciones (Morán, 2018). Se utiliza una CNN para comprender imágenes individuales ya que pueden reconocer y clasificar características particulares de las mismas. Una red neuronal recurrente (RCNN) se usa de manera similar

para aplicaciones de video para ayudar a las computadoras a comprender cómo se relacionan entre sí las imágenes en una serie de cuadros (Kim & Reiter, 2017).

Figura 8

Esquema de una red neuronal convolucional.



Fuente: (Li et al., 2021)

En la Figura 8, se observa el esquema de una red neuronal convolucional, que muestra dos capas. Cada capa consta de varios planos bidimensionales de neuronas, una capa por canal de salida. Una neurona dada en la capa k recibe información de un parche de la capa inferior, cada entrada ponderada por un conjunto de coeficientes del núcleo. Cada neurona en la misma capa suele compartir el mismo conjunto de coeficientes, aunque los diferentes canales de salida tienen núcleos distintos (Li et al., 2021).

2.5.2. Visión Artificial

La visión artificial o visión por computador es un campo de la inteligencia artificial que permite que las computadoras y los sistemas obtengan información significativa de imágenes digitales, videos u otras entradas visuales, y tomen medidas o hagan recomendaciones basadas en esa información. Si la IA permite que las

computadoras piensen, la visión por computadora les permite ver, observar y comprender (IBM, 2012b).

De acuerdo con la Asociación de Imágenes Automatizada (AIA) generalmente los recursos utilizados en la visión artificial son más exigentes en cuestiones de solidez, confiabilidad y estabilidad debido a que los resultados obtenidos de los sensores digitales integrados en camaras, son procesados y analizados en un software de visión que como ya se mencionó anteriormente, identifican detalles importantes de objetos, generando confiabilidad en los resultados (COGNEX, 2016).

La visión artificial necesita muchos datos. Ejecuta análisis de datos una y otra vez hasta que discierne distinciones y finalmente reconoce imágenes. Por ejemplo, para entrenar a una computadora para que reconozca llantas de automóviles, necesita recibir grandes cantidades de imágenes de llantas y elementos relacionados con llantas para aprender las diferencias y reconocer una llanta, especialmente una sin defectos. Se utilizan dos tecnologías esenciales para lograr esto: un tipo de aprendizaje automático llamado aprendizaje profundo (deep learning) y una red neuronal convolucional (CNN) (Basheer & Hajmeer, 2000).

2.5.2.1.Estimación De Pose Humana.

La estimación de poses humana es una técnica de visión por computadora que predice diferentes poses en función de las partes del cuerpo de una persona y la posición de las articulaciones en una imagen o video (Song et al., 2021). Esto generalmente se hace identificando, ubicando y rastreando una cantidad de puntos clave en un objeto o persona determinada. Por tanto, el modelo estimador de pose recibe una imagen o video como

entrada y genera las coordenadas de las articulaciones y partes del cuerpo que detecta (Cao et al., 2017).

En la Figura 9 se puede observar el resultado obtenido al ensamblar las partes del cuerpo humano que se han detectado previamente mediante una CNN de todas las personas en la imagen.

Figura 9

Estimación de pose humana.

Fuente: (Cao et al., 2017)

En los últimos años, se han desarrollado numerosos conjuntos de datos para la evaluación de algoritmos de estimación de poses humanas en 2D. Sin embargo, los conjuntos de datos en 3D son menos variados, en gran parte debido a las restricciones impuestas por los sensores utilizados para la captura de posturas en tres dimensiones (Song et al., 2021).

2.5.2.1.1. *Estimación De Pose Humana 2D.*

La estimación de pose 2D simplemente estima la ubicación de los puntos clave en el espacio 2D en relación con una imagen o un cuadro de video. El modelo estima una coordenada X e Y para cada punto clave (Andriluka et al., 2014).

2.5.2.1.2. *Estimación De Pose Humana 3D.*

La estimación de pose 3D funciona para transformar un objeto en una imagen 2D en un objeto 3D agregando una dimensión z a la predicción. Nos permite predecir el posicionamiento espacial real de una persona u objeto representado (Cao et al., 2017).

Finalmente, existe una distinción entre detectar uno o múltiples objetos en una imagen o video. Estos dos enfoques se pueden denominar estimación de pose única y múltiple, y en gran medida se explican por sí mismos: los enfoques de estimación de pose única detectan y rastrean una persona u objeto, mientras que los enfoques de estimación de pose múltiple detectan y rastrean varias personas u objetos (Odemakinde, 2023).

2.5.2.1.3. *Enfoques Para La Estimación de Pose.*

En general, las arquitecturas de aprendizaje profundo adecuadas para la estimación de poses se basan en variaciones de redes neuronales convolucionales (CNN). Existen dos enfoques generales: un enfoque de abajo hacia arriba y un enfoque de arriba hacia abajo.

- **Enfoque de abajo hacia arriba:** Con este enfoque, el modelo detecta cada instancia de un punto clave en particular (por ejemplo, todas las

manos izquierdas) en una imagen dada y luego intenta ensamblar grupos de puntos clave en esqueletos para distintos objetos(Dubey & Dixit, 2023).

- **Enfoque de arriba hacia abajo:** Con este enfoque, la red primero usa un detector de objetos para dibujar un cuadro alrededor de cada instancia de un objeto y luego estima los puntos clave dentro de cada región recortada (Song et al., 2021).

2.5.2.2.Métodos De Estimación De Pose.

Debido a que la estimación de pose es una técnica de visión por computadora fácilmente aplicable, podemos implementar un estimador de pose personalizado utilizando arquitecturas existentes. Entre los principales métodos de estimación de pose humana tenemos los siguientes:

OpenPose. Esta arquitectura presenta una estimación de pose de varias personas en tiempo real. OpenPose es una detección de múltiples personas en tiempo real de código abierto, con alta precisión en la detección de puntos clave del cuerpo, el pie, la mano y la cara. Una ventaja de OpenPose es que es una API que brinda a los usuarios la flexibilidad de seleccionar imágenes de origen desde campos de cámaras, cámaras web y otros, más importante aún para aplicaciones de sistemas integrados (por ejemplo, integración con cámaras y sistemas de CCTV). Admite diferentes arquitecturas de hardware, como GPU CUDA, GPU OpenCL o dispositivos solo de CPU (Cao et al., 2017).

High-Resolution Net (HRNet). Es una arquitectura utilizada en problemas de procesamiento de imágenes para encontrar lo que conocemos como puntos clave (articulaciones) con respecto al objeto o persona específica en una imagen. Una ventaja de esta arquitectura sobre otras arquitecturas es que la mayoría de los métodos existentes

combinan representaciones de posturas de alta resolución a partir de representaciones de baja resolución con respecto al uso de redes de alta y baja resolución. En lugar de este sesgo, la red neuronal mantiene representaciones de alta resolución al estimar las posturas. Por ejemplo, esta arquitectura HRNet es útil para la detección de la postura humana en los deportes televisados (Odemakinde, 2023).

AlphaPose. Es útil para detectar poses en presencia de cuadros delimitadores humanos inexactos. Es decir, es una arquitectura óptima para estimar poses humanas a través de cuadros delimitadores detectados de manera óptima. La arquitectura AlphaPose es aplicable para detectar poses de una o varias personas en imágenes o campos de video (Song et al., 2021).

DensePose. Esta es una técnica de estimación de pose que tiene como objetivo mapear todos los píxeles humanos de una imagen RGB en la superficie 3D del cuerpo humano. DensePose también se puede usar para problemas de estimación de una o varias poses (Dubey & Dixit, 2023).

OpenPifPaf. Es una biblioteca de visión por computadora de código abierto y un marco para la estimación de poses, que implica identificar y localizar partes del cuerpo humano en imágenes o videos. Está construido sobre el marco de aprendizaje profundo de PyTorch y utiliza un enfoque de aprendizaje de tareas múltiples para lograr una estimación de pose precisa y eficiente. OpenPifPaf ha ganado popularidad por su facilidad de uso, robustez y capacidad para manejar escenarios desafiantes de estimación de poses, como oclusión y fondos desordenados (Odemakinde, 2023).

2.6.Software libre

Para desarrollar el sistema de visión artificial planteado es necesario utilizar diferentes herramientas de software libre, tales como:

2.6.1. Python

Python es un lenguaje de programación de alto nivel interpretado, orientado a objetos y con semántica dinámica. Sus estructuras de datos integradas de alto nivel, combinadas con la escritura y el enlace dinámicos, lo hacen muy atractivo para el desarrollo rápido de aplicaciones, así como para su uso como lenguaje de secuencias de comandos o pegamento para conectar componentes existentes entre sí (Rawat Ajay, 2020). La sintaxis simple y fácil de aprender de Python enfatiza la legibilidad y, por lo tanto, reduce el costo de mantenimiento del programa. Python admite módulos y paquetes, lo que fomenta el modularidad del programa y la reutilización del código. El intérprete de Python y la extensa biblioteca estándar están disponibles en formato fuente o binario sin cargo para todas las plataformas principales y se pueden distribuir gratuitamente (PYTHON, 2015).

Python es un lenguaje de programación versátil que se puede ejecutar en múltiples sistemas operativos, como Windows, Linux, UNIX, Mac OS, entre otros. Su capacidad multiplataforma permite trasladar programas de una plataforma a otra sin necesidad de realizar modificaciones. Aunque los derechos del lenguaje están reservados para el instituto Python, es un software de código abierto, lo que significa que no existen restricciones para su uso, modificación y distribución. No solo es posible desarrollar y compartir software escrito en Python, sino también realizar cambios en su código fuente (Srinath, 2017).

Por estas razones, la popularidad de Python ha ido expandiéndose en áreas como la ciencia de datos y el aprendizaje automático de una manera acelerada en los últimos años.

2.6.2. OpenCv

OpenCV (Biblioteca de visión artificial de código abierto) es una biblioteca de software de aprendizaje automático y visión artificial de código abierto. OpenCV se creó para proporcionar una infraestructura común para las aplicaciones de visión por computadora y para acelerar el uso de la percepción de la máquina en los productos comerciales. Lo que diferenció desde un principio a OpenCV de otros paquetes de visión por computador, fue su constante actualización de imágenes y el área de trabajo para trabajar con dichas imágenes de manera dinámica y estática, ya que al ser compatible con la mayoría de las capturadoras/cámaras del mercado y los sistemas operativos más conocidos como Windows y Linux de ese entonces, fue posible su evolución a lo que conocemos hoy en día. (Arévalo-Espejo et al., 2005).

Al ser un producto con licencia de Apache 2, OpenCV facilita que las empresas utilicen y modifiquen el código. La biblioteca posee una gran cantidad de algoritmos optimizados, que incluyen un conjunto completo de algoritmos de aprendizaje automático y visión por computadora clásicos y de última generación. Estos algoritmos se pueden usar para detectar y reconocer rostros, identificar objetos, clasificar acciones humanas en videos, rastrear movimientos de cámara, rastrear objetos en movimiento, extraer modelos 3D de objetos, producir nubes de puntos 3D de cámaras estéreo, unir imágenes para producir una alta resolución. imagen de una escena completa, encontrar imágenes similares de una base de datos de imágenes, eliminar los ojos rojos de las imágenes tomadas con flash, seguir los movimientos de los ojos, reconocer el escenario y establecer

marcadores para superponerlo con la realidad aumentada, entre otros. OpenCV tiene más de 47 mil personas de usuario comunidad y número estimado de descargas que superan los 18 millones. La biblioteca se utiliza ampliamente en empresas, grupos de investigación y organismos gubernamentales (OPENCV, 2018).

2.7.IEEE 29148

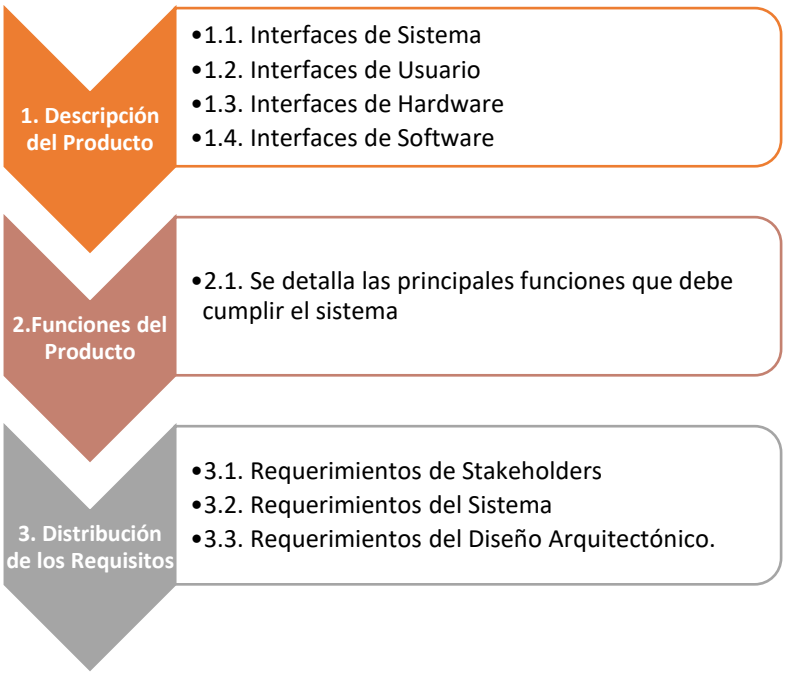
Este estándar especifica los procesos requeridos implementados en las actividades de ingeniería que dan como resultado requisitos para sistemas y productos de software (incluidos los servicios) a lo largo del ciclo de vida (ISO/IEC/IEEE, 2011).

La ingeniería de requisitos es una función interdisciplinaria que media entre los dominios del adquirente y del proveedor para establecer y mantener los requisitos que debe cumplir el sistema, software o servicio de interés. El resultado de la ingeniería de requisitos es una jerarquía de requisitos que: permite un entendimiento acordado entre las partes interesadas (por ejemplo, adquirentes, usuarios, clientes, operadores, proveedores), se valida frente a las necesidades del mundo real, se puede implementar y proporciona una base para verificar diseños y aceptar soluciones (ISO/IEC/IEEE, 2011).

A continuación, en la Figura 10 se detalla brevemente el contenido del esquema de especificación de requerimientos del estándar IEEE 29148.

Figura 10

Resumen de especificaciones del estándar IEEE 29148.



Fuente: Adaptado de (ISO/IEC/IEEE, 2011)

Descripción del producto: En esta etapa se procede a definir la relación del sistema con otros productos relacionados, para esto se selecciona la o las interfaces que sean necesarias, entre las principales interfaces tenemos:

- **Interfaces del sistema**: Se enumera cada interfaz del sistema e identifica la funcionalidad del software para lograr que el requisito y la descripción de la interfaz coincidan con el sistema (ISO/IEC/IEEE, 2011).
- **Interfaces de usuario:** Se debe especificar lo siguiente:

  a) Las características lógicas de cada interfaz entre el producto de software y sus usuarios. Esto incluye aquellas características de configuración, por ejemplo, formatos de pantalla requeridos, diseños de páginas o ventanas,

contenido de cualquier informe o menú, disponibilidad de teclas de función programables.

b) Todos los aspectos de la optimización de la interfaz con la persona que utiliza o proporciona otro tipo de apoyo al sistema. Esto simplemente puede comprender una lista de cosas que hacer y qué no hacer en cómo el sistema aparecerá al usuario. Un ejemplo puede ser un requisito para la opción de mensajes de error largos o cortos (ISO/IEC/IEEE, 2011).

- **Interfaces de hardware:** Especificar las características lógicas de cada interfaz entre el producto de software y los elementos de hardware del sistema. Esto incluye características de configuración (número de puertos, conjuntos de instrucciones, dispositivos a utilizar) (ISO/IEC/IEEE, 2011).
- **Interfaces de software:** Especificar el uso de otros productos de software requeridos (por ejemplo, un sistema de gestión de datos, un sistema operativo, o un paquete matemático), o que interactúa con otros sistemas de aplicación (por ejemplo, el enlace entre un sistema de cuentas por cobrar y un sistema de contabilidad general) (ISO/IEC/IEEE, 2011).

Funciones del producto: Se debe proporcionar un resumen de las principales funciones que el software realizará. Por ejemplo, una Liquidación de Recibos Evaluados traducido del inglés Evaluated Receipt Settlement (ERS) para un programa de contabilidad puede usar esta parte para tratar el mantenimiento de cuentas de clientes, la declaración de clientes y la preparación de facturas sin mencionar la gran cantidad de detalle que cada una de esas funciones requiere (ISO/IEC/IEEE, 2011).

Distribución de los requisitos: Para requisitos que requieran la implementación sobre múltiples elementos de software, o cuando la asignación a un elemento de software

es inicialmente indefinida, esto debe ser dicho. Se utilizará una tabla de referencia cruzada por función y elemento de software para resumir la distribución.

- **Requerimientos de Stakeholders**: Los requisitos de las partes o stakeholders son las necesidades, expectativas y limitaciones definidas por las partes interesadas para un proyecto, producto o sistema. Estos requisitos ayudan a guiar el proceso de desarrollo para garantizar que el resultado final cumpla con las expectativas de las partes interesadas (ISO/IEC/IEEE, 2011).
- **Requerimientos del Sistema:** Estos requisitos definen las especificaciones, condiciones y capacidades necesarias que un sistema debe cumplir para funcionar como se espera. Estos requisitos proporcionan una base para diseñar, desarrollar y probar un sistema para asegurar que cumpla con las necesidades del negocio y de las partes interesadas (ISO/IEC/IEEE, 2011).
- **Requerimientos del Diseño Arquitectónico:** El propósito del proceso de diseño arquitectónico es sintetizar una solución que satisfaga los requisitos del sistema (ISO/IEC/IEEE, 2011).

3. Diseño del Sistema

En este capítulo se examina en detalle la situación actual que da origen al proyecto, identificando los problemas, necesidades y oportunidades que justifican su desarrollo. Además, se describe la metodología utilizada para su diseño e implementación, especificando las fases, enfoques y técnicas aplicadas. También se justifica la selección de los componentes de hardware y software, considerando criterios de compatibilidad, eficiencia y rendimiento. Finalmente, se abordan todos los elementos

esenciales para asegurar un diseño sólido y una implementación efectiva del sistema, garantizando su funcionalidad y cumplimiento de los objetivos planteados.

3.1. Situación Actual

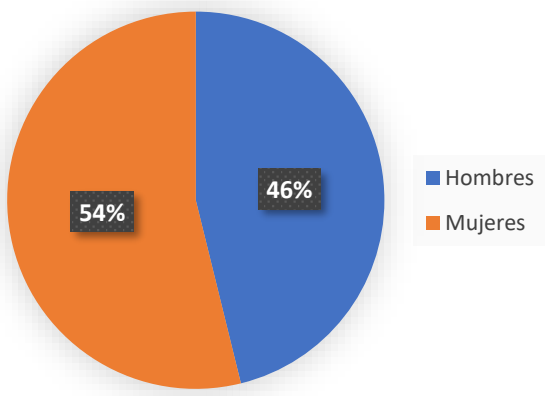
La medición de ángulos articulares en movimientos cervicales es evaluado manualmente, con herramientas como lo son graduadores o goniómetros que proporcionan cierto error de medición por estimación visual propia del ser humano, es así que toma importancia la inclusión de la tecnología para determinar el avance en la rehabilitación de movilidad cervical. Para realizar las pruebas del sistema, se acudió al asilo León Ruales en la ciudad de Ibarra.

El Centro Gerontológico Residencial León Ruales colabora con la Universidad Técnica del Norte en la formación de estudiantes que cursan los últimos niveles de la carrea de Fisioterapia. El objetivo de esta alianza es capacitar a futuros profesionales en el diseño de programas de rehabilitación física y ejercicios dirigidos a mejorar la agilidad, fuerza y capacidad cardiovascular de los adultos mayores, permitiéndoles realizar sus actividades diarias de manera autónoma con menor riesgo de caídas o problemas osteoarticulares.

Actualmente, en el periodo académico octubre 2024 - febrero 2025, el Centro Gerontológico Residencial León Ruales alberga a un total de 39 pacientes, de los cuales 18 son hombres (46% del total) y 21 son mujeres (54% del total). La distribución de estos datos se presenta en la Figura 11.

Figura 11

Total de pacientes albergados en el asilo



Fuente: Autoría

El estudio llevado a cabo en esta sección respalda la implementación del presente proyecto como una herramienta tecnológica para la evaluación de pacientes. Además, permite gestionar una base de datos personalizada para cada paciente y ofrecer un sistema interactivo que contribuya a su recuperación. A través de evaluaciones precisas, el sistema proporcionará datos fiables y en menor tiempo comparado al sistema actual.

3.2.Metodología

Para el desarrollo del presente proyecto se empleará la metodología denominada “Modelo en V”, diseñada para garantizar y mejorar la calidad del software, optimizando recursos en cada una de sus etapas. Este modelo establece una relación directa entre cada fase del desarrollo y su correspondiente fase de prueba, lo que permite validar y verificar el correcto funcionamiento del sistema en cada etapa del proceso.

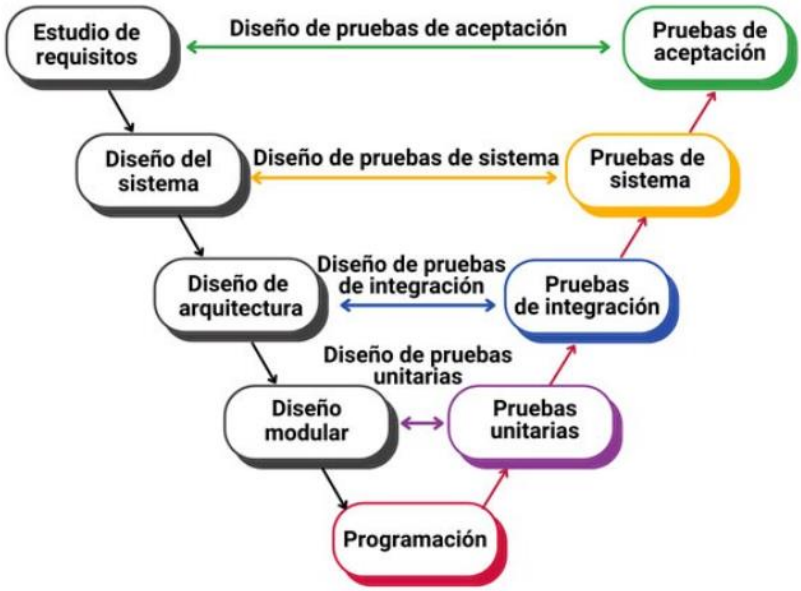
En la Figura 12 se observa que la parte izquierda de la "V" representa las etapas de desarrollo, que incluyen la iniciación del proyecto, la especificación de requisitos, el

diseño preliminar y la codificación del software. Por otro lado, en la parte derecha de la "V" se encuentran las pruebas de verificación y validación, las cuales corresponden a cada una de las etapas del desarrollo del sistema.

Esta metodología facilita significativamente la gestión del proyecto gracias a la estrecha relación entre cada nivel y los distintos tipos de pruebas, lo que permite detectar y corregir fallos en cada una de las etapas del desarrollo, hasta su respectiva culminación.

Figura 12

Modelo en V



Fuente: De (Herrera, 2010)

El primer nivel de este modelo (Estudio de requisitos) establece los parámetros iniciales y los requisitos para el desarrollo del proyecto, para esto, es necesario definir las funciones que el sistema deberá ejecutar y considerar los Stakeholders que formarán parte del proceso (Herrera, 2010).

El segundo nivel (Diseño del sistema), se enfoca en definir la solucion a los requisitos previamente establecidos. En esta etapa, se selecciona la tecnología adecuada para abordar el problema planteado al inicio del proyecto. Para ello, se consideran las

funciones del sistema, los elementos de la interfaz de usuario y la estructura de los datos que serán procesados y medidos (Herrera, 2010).

El tercer nivel (Diseño de arquitectura), define los componentes de software y hardware que proporcionan una perspectiva sobre el diseño del proyecto, así como la arquitectura del sistema. Además, se realizan pruebas de integración para evaluar cada etapa del proyecto y garantizar que cumpla con las expectativas establecidas al inicio (Herrera, 2010).

Los niveles cuarto y quinto corresponden a la fase de implementación, desarrollo y programación del software. En esta etapa, se traducen los diseños y especificaciones en código funcional, integrando los distintos módulos y componentes del sistema. Además, se llevan a cabo pruebas preliminares para asegurar que el sistema opere conforme a lo previsto, garantizando su estabilidad, rendimiento y cumplimiento con los requerimientos establecidos (Herrera, 2010).

Definición de términos abreviados del estándar ISO/IEC/IEEE 29148:2011

Para el diseño del sistema, se tomó como referencia el estándar ISO/IEC/IEEE 29148:2011, el cual proporciona directrices para la ingeniería de requisitos. Este estándar internacional establece procesos para la definición, planificación e implementación de requisitos, brindando orientación adicional a los usuarios.

Entre los principales procesos definidos en el estándar se encuentran:

1. Proceso de definición de requisitos de stakeholders
2. Proceso de análisis de requisitos, basado en ISO/IEC 15288 para sistemas en general o ISO/IEC 12207 para sistemas de software.

En el desarrollo del presente proyecto se van a utilizar algunos de los términos y definiciones dados en ISO/IEC/IEEE 29148. La tabla 1 muestra las abreviaturas de cada tipo de requerimiento que se va a utilizar.

Tabla 1

Definición de Abreviaturas

Abreviatura	Descripción
StRS	Requerimientos de Stakeholder
SyRS	Requerimientos del Sistema
DaRS	Requerimientos de Diseño Arquitectónico

Fuente: Autoría

Por otro lado, es importante destacar la prioridad de cada requerimiento, la cual se especificará según lo indicado en la Tabla 2.

Tabla 2

Prioridad de los Requerimientos del sistema

Prioridad	Descripción
Alta	Aquellos requerimientos críticos que se deberán incluir en todo momento para que no afecte la funcionalidad del sistema.
Media	La falta de estos requerimientos puede afectar la decisión final del sistema, sin embargo, se pueden omitir en condiciones de fuerza mayor.
Baja	No se espera un impacto significante en caso de no incluir este tipo de requerimiento.

Fuente: Adaptado de (Pérez García, 2013)

3.3.Requerimientos de Stakeholders

Los requerimientos de stakeholders son fundamentales para identificar las necesidades, expectativas y restricciones de todas las partes interesadas en el proyecto. Estos requisitos orientan el desarrollo del sistema, asegurando que el producto final cumpla con los objetivos del negocio y las necesidades de los usuarios.

En la Tabla 3 se muestra la lista de stakeholders considerados en el desarrollo del proyecto, mientras que en la Tabla 4 se observa los requerimientos de los stakeholders.

Tabla 3

Lista de Stakeholders

Lista de Stakeholders
1. Adultos mayores (Pacientes a ser evaluados con el sistema)
2. Msc. Edgar Maya (director del trabajo de titulación)
3. Msc. Ana Umaquinga (codirectora del trabajo de titulación)
4. Msc. Daniela Zurita (profesional de salud, usuaria del sistema)
5. Stalin Ibarra (desarrollador del proyecto)

Fuente: Autoría

Tabla 4

Requerimientos de stakeholders

	REQUERIMIENTOS DE STAKEHOLDERS (StRS)	**PRIORIDAD**		
		Alta	Media	Baja
StRS1	El sistema se debe implementar en un lugar fijo y con buena iluminación.	X		
StRS2	La cámara del sistema debe estar a una distancia determinada.	X		
StRS3	La interfaz de usuario del sistema debe ser amigable y comprensiva para el usuario.	X		
StRS4	El sistema no debe generar molestias hacia los pacientes.	X		

StRS5	El procesamiento de las imágenes del sistema debe ser rápida.	X	
StRS6	El informe diagnóstico generado por el sistema debe ser claro y legible.		X

Fuente: Autoría

3.4.Requerimientos del Sistema

Se procede a establecer los requisitos del sistema, definiendo los límites funcionales en cuanto a su comportamiento y las propiedades que se deben proporcionar. Estos requisitos especifican las condiciones, características y capacidades que el sistema debe cumplir para asegurar su funcionamiento adecuado.

3.4.1. Requerimientos iniciales del sistema

En los requisitos iniciales del sistema se define cada función que debe cumplir, especificando la calidad esperada del rendimiento, incluidos los operadores necesarios para ejecutar dicha función. Además, se detallan las condiciones bajo las cuales el sistema debe ser capaz de llevar a cabo la función, cuándo debe iniciarla y en qué circunstancias debe dejar de ejecutarla. En la Tabla 5 se muestran los requerimientos iniciales del sistema (SyRS).

Tabla 5

Requerimientos del sistema

	SYSR			
	REQUERIMIENTOS DEL SISTEMA			
		Prioridad		
		Alta	**Media**	**Baja**
	Requerimientos de Interfaz			
SyRS1	El sistema necesita estar conectado a la red eléctrica del centro de rehabilitación.	X		
SyRS2	El sistema interactúa por medio de una cámara web entre usuario y paciente.	X		

SyRS3	El sistema debe ser capaz de detectar las distintas articulaciones del raquis cervical.	X		
SyRS4	El sistema debe obtener las mediciones en tiempo real.	X		
SyRS5	El sistema de almacenar los ángulos obtenidos de los movimientos cervicales.	X		
	Requerimientos de Performance			
SyRS6	Reconocimiento de cabeza, cuello, nariz y hombros para el cálculo de ángulos cervicales.	X		
SyRS7	Permitir y guardar mas de un diagnóstico por paciente.		X	
SyRS8	Detección de articulaciones de miembros inferiores.			X
SyRS9	Obtención de video en tiempo real de la cámara web	X		
	Requerimientos Físicos			
SyRS10	El sistema debe estar situado en un lugar donde no cause molestias o interferencias entre los diferentes stakeholders.	X		
SyRS11	La cámara debe situarse a una cierta distancia del paciente y con una iluminación adecuada.		X	
SyRS12	El sistema deberá tener la ventilación adecuada para evitar sobre calentamientos.		X	

Fuente: Autoría

3.5.Requerimientos de Diseño Arquitectónico

En los requerimientos de diseño arquitectónico se especifican de manera detallada los requerimientos de hardware y software necesarios para la implementación del proyecto. El objetivo fundamental del proceso de diseño arquitectónico es sintetizar una solución que no solo cumpla con los requisitos del sistema, sino que también garantice la eficiencia, escalabilidad y sostenibilidad a largo plazo.

El diseño arquitectónico se define en términos de los requisitos del conjunto de elementos que conforman el sistema, considerando cada uno de sus componentes, sus interacciones y el comportamiento esperado. Esto incluye tanto los componentes físicos como los virtuales, y es esencial asegurar que todos los aspectos del sistema estén alineados con los objetivos de rendimiento y funcionalidad.

Se define una solución de diseño arquitectónico en términos de los requisitos para el conjunto de elementos del sistema de los cuales está configurado el sistema. Es importante establecer y mantener la trazabilidad entre los requisitos y el diseño arquitectónico, incluidos los elementos e interfaces del sistema.

Las directrices específicas descritas en los requerimientos de arquitectura, que se presentan en la Tabla 6, son fundamentales para orientar la selección del hardware y software adecuados para el proyecto. Estas directrices no solo determinan las especificaciones técnicas, sino que también tienen en cuenta factores como la compatibilidad, la interoperabilidad y la capacidad de adaptación del sistema a futuras necesidades.

Tabla 6

Requerimientos del Diseño Arquitectónico

#	REQUERIMIENTOS	Prioridad		
		Alta	**Media**	**Baja**
	Requerimientos de Diseño			
DaRS1	La cámara debe estar situada en el mismo lugar para todas las mediciones.	X		
DaRS2	Los cables de conexiones tanto eléctricas como de datos no deben interferir en el sistema	X		
DaRS3	El sistema debe encontrase en un lugar fijo y firme paralelo al suelo.		X	
	Requerimientos de Hardware			
DaRS4	El sistema requiere una unidad central de procesamiento (CPU) con la capacidad necesaria (núcleos) para procesar imagen y video.	X		
DaRS5	El sistema requiere una unidad de procesamiento grafico (GPU) con tecnología de procesamiento de cómputo paralelo CUDA.	X		
DaRS6	Se requiere una cámara web que permita grabar video en buena calidad y sea compatible con el sistema.	X		

DaRS7	El sistema necesita de un periférico de almacenamiento para guardar todos los diagnósticos.	X	
	Requerimientos de Software		
DaRS8	Se requiere de un sistema operativo y lenguaje de programación de código abierto	X	
DaRS9	Se requiere compatibilidad con la librería OpenCV y la cámara	X	
DaRS10	Se requiere que el software permita ejecutar el código de visión artificial en tiempo real en el servidor	X	
DaRS11	Se requiere de un sistema operativo que ejecute con rapidez los hilos de procesamiento del sistema	X	
DaRS12	Se requiere software que permita usar los recursos de la GPU en el uso de modelos de visión artificial	X	
DaRS13	Se requiere de un software de base de datos no relacional que permita obtener un bajo impacto o costo en el rendimiento del sistema	X	
DaRS14	Se requiere de un software de diseño de interfaces de usuario (GUI) para la visualización de resultados		X

Fuente: Autoría

3.6.Selección de hardware y software

La selección de los componentes de hardware y software es un paso fundamental en el desarrollo del sistema, ya que influye directamente en su desempeño, estabilidad y escalabilidad. Para garantizar una elección óptima, se sigue un proceso estructurado basado en criterios técnicos y requisitos específicos.

Se establecen los criterios clave para la evaluación de los componentes, los cuales se basan en:

- Requerimientos de los Stakeholders (StRS): Necesidades y expectativas de los interesados en el sistema.

- Requerimientos del Sistema (SySR): Especificaciones técnicas y operativas necesarias para el correcto funcionamiento.
- Requerimientos de Diseño Arquitectónico (DaRS): Directrices que aseguran la integración eficiente de hardware y software dentro de la arquitectura del sistema.

Luego, se elabora una tabla comparativa en la que se listan las opciones de hardware y software disponibles junto con sus especificaciones técnicas y atributos evaluados.

Cada componente es analizado según su cumplimiento con los criterios establecidos. Se asignan puntuaciones a cada atributo relevante, considerando aspectos como rendimiento, compatibilidad, costo, escalabilidad, eficiencia energética y soporte técnico.

El componente que obtenga la puntuación más alta en la evaluación global es seleccionado, asegurando que cumpla con los requisitos funcionales y estratégicos del sistema. Este proceso permite tomar decisiones fundamentadas, minimizando riesgos y optimizando el rendimiento del sistema.

3.6.1. Hardware

En el apartado anterior, se presentaron tablas con los distintos tipos de requisitos para el diseño e implementación del sistema. En la Tabla 6, se detallan específicamente los requerimientos para el diseño arquitectónico, con un enfoque particular en los criterios que deben cumplir los elementos de hardware.

A partir de esta información, se procede a seleccionar la unidad central de procesamiento (CPU), la unidad de procesamiento de gráficos (GPU) y la cámara web,

asegurando que cumplan con las especificaciones necesarias para la captura de imágenes y el correcto funcionamiento del sistema.

3.6.1.1.CPU.

La documentación de OpenPose especifica los requisitos mínimos de hardware necesarios para el correcto funcionamiento de esta biblioteca de estimación de pose humana. En ella se indica que se requiere un procesador de al menos 4 núcleos, siendo lo óptimo o más recomendable un procesador de 8 núcleos para un rendimiento óptimo.

Con base en estos criterios, se han seleccionado tres opciones de procesadores que mejor se ajustan a las necesidades del proyecto. En la Tabla 7 se presentan las evaluaciones de cada procesador, y a partir de esta comparación, se elige la opción más adecuada para el sistema.

Tabla 7

Comparación entre distintos procesadores

Procesador	**Requerimientos**						**Valoración Total**
	DaRS4	**DaRS8**	**DaRS10**	**DaRS11**	**DaRS13**	**DaRS14**	
Intel Core i5-13600KF	SI	SI	SI	SI	SI	SI	6
AMD Ryzen 7 5800X3D	SI	SI	SI	SI	SI	SI	6
Intel Core i5- 8300H	SI	SI	SI	NO	SI	SI	5

Fuente: Autoría

A partir de la Tabla 7, se concluye que las dos primeras opciones serían las más adecuadas para el desarrollo del proyecto. Sin embargo, se opta por la tercera opción, ya que este procesador está disponible y cumple con la mayoría de las funciones requeridas

por el sistema. Por lo tanto, se selecciona el procesador **Intel Core i5-8300H**, en la Tabla 8 se pueden observar sus características más relevantes:

Tabla 8

Características del procesador seleccionado.

Especificaciones del Procesador Intel Core i5 8300H	
Cantidad de núcleos	4
Total de subprocesos	8
Frecuencia máxima	4.00 GHz
Frecuencia básica	2.30GHz

Fuente: Adaptado de (Intel, 2010)

3.6.1.2.GPU.

Para la selección del procesador gráfico, se compararon dos tarjetas de video considerando que el requisito mínimo es una GPU con al menos 1.5 GB de memoria disponible. La Tabla 9 presenta la evaluación de cada tarjeta en función de los requisitos, permitiendo determinar la opción más adecuada para el proyecto.

Tabla 9

Comparación entre tarjetas de video.

GPU	**Requerimientos**			**Valoración Total**
	DaRS5	**DaRS12**	**DaRS10**	
AMD RADEON RX 480	NO	NO	SI	1
NVIDIA GeForce GTX 1050	SI	SI	SI	3

Fuente: Autoría

A partir de la Tabla 9, se concluye que la primera opción sería las más adecuadas para el desarrollo del proyecto. Además, se opta por la misma, ya que este procesador de gráficos está disponible y cumple con las funciones requeridas por el sistema. Por lo tanto,

se selecciona el procesador de gráficos **NVIDIA GeForce GTX1050**, en la Tabla 10 se pueden observar sus características más relevantes:

Tabla 10

Características del procesador de gráficos seleccionado.

Especificaciones de la Tarjeta Gráfica NVIDIA GeForce GTX 1050	
NVIDIA CUDA®Cores	768
Velocidad de Memoria	7 Gbps
Configuración de Memoria Estándar	4 GB GDDR5
Ancho de Interfaz de Memoria	128-bit

Fuente: Adaptado de (NVIDIA, 2008)

3.6.1.3.Cámara web.

Para la selección de la cámara web, se compararon dos tipos de cámaras, las cuales cumplen con las necesidades del sistema. La Tabla 11 presenta la evaluación de cada cámara en función de los requisitos, permitiendo determinar la opción más adecuada para el proyecto.

Tabla 11

Comparación entre cámaras web.

CÁMARA WEB	**Requerimientos**		**Valoración Total**
	DaRS6	**DaRS9**	
Trust Trino HD	SI	NO	1
Logitech HD webcam C270	SI	SI	2

Fuente: Autoría

A partir de la Tabla 11, se concluye que la segunda opción sería las más adecuadas para el desarrollo del proyecto. Además, se opta por la misma, ya que este procesador de

gráficos está disponible y cumple con las funciones requeridas por el sistema. Por lo tanto, se selecciona la cámara web **Logitech HD webcam C270**, en la Tabla 12 se pueden observar sus características más relevantes:

Tabla 12

Características de la cámara web escogida.

Especificaciones de la cámara web Logitech HD webcam C270
Videoconferencias HD 720p en Pantalla Panorámica
Nítidas Fotos de 3 MP
Compatible con: Windows 7, Windows 8, Windows 10, MacOS 10.10 o superior y ChromeOS

Fuente: Autoría

3.6.2. Software

Para la selección del software, se debe tener en cuenta todos los requerimientos mínimos de OpenPose, los cuales son (OpenPose, 2019):

- Sistema operativo: Ubuntu (14 o 16) o Windows (probado en 10, 8 y 7).
- GPU con al menos 1,5 GB y 4GB de memoria RAM
- CUDA y cuDNN instalados.
- Muy recomendable: una CPU con al menos 8 núcleos.

Como se puede observar, este modelo de aprendizaje automático debe ser compatible con la tecnología de procesamiento paralelo CUDA y con la biblioteca de visión artificial OpenCV. Además, es fundamental asegurarse de que cumpla con los requisitos de software especificados en la Tabla 6.

Por lo tanto, la selección del software de programación requiere especial atención. Para este fin, se evaluaron cuatro opciones que se ajustan parcialmente a los requisitos del proyecto para su desarrollo. La Tabla 13 muestra la valoración de cada opción en función de dichos requerimientos, lo que permite determinar el software más adecuado.

Tabla 13

Comparación de software de programación.

Software	Requerimientos							Valoración Total
	DaRS8	DaRS9	DaRS10	DaRS11	DaRS12	DaRS13	DaRS14	
MatLab	SI	SI	SI	NO	SI	SI	SI	6
Java	SI	SI	SI	SI	NO	SI	SI	6
Python	SI	SI	SI	SI	SI	SI	SI	7
C++	SI	SI	SI	SI	SI	SI	NO	6

Fuente: Autoría

Como se puede observar en la Tabla 13, el software más adecuado para la realización del proyecto es **Python.** Además, Python es un lenguaje de programación que se utiliza ampliamente en el campo de la inteligencia artificial (IA) debido a que su sintaxis concisa y legible facilita la lectura y escritura de código.

3.7.Diseño del Sistema

Una vez analizados los requisitos del sistema, junto con las necesidades de los stakeholders y los distintos componentes que lo conformarán, se procede con su implementación. Para ello, se diseña la arquitectura del sistema, asegurando que cada elemento se integre de manera eficiente y cumpla con los objetivos planteados.

Como parte de este proceso, se elabora un diagrama de bloques que describe detalladamente cada una de las etapas del sistema, desde la adquisición de datos hasta el procesamiento y la interacción con los usuarios. Este enfoque permite visualizar la estructura funcional del sistema y comprender cómo se interconectan sus distintos módulos.

Además, el diseño y la planificación garantizan que el sistema opere de manera óptima, cumpliendo con los requisitos tanto de los usuarios como de los pacientes. De este modo, se busca ofrecer un entorno confiable, eficiente y adaptado a las necesidades específicas del proyecto.

3.7.1. Arquitectura del sistema

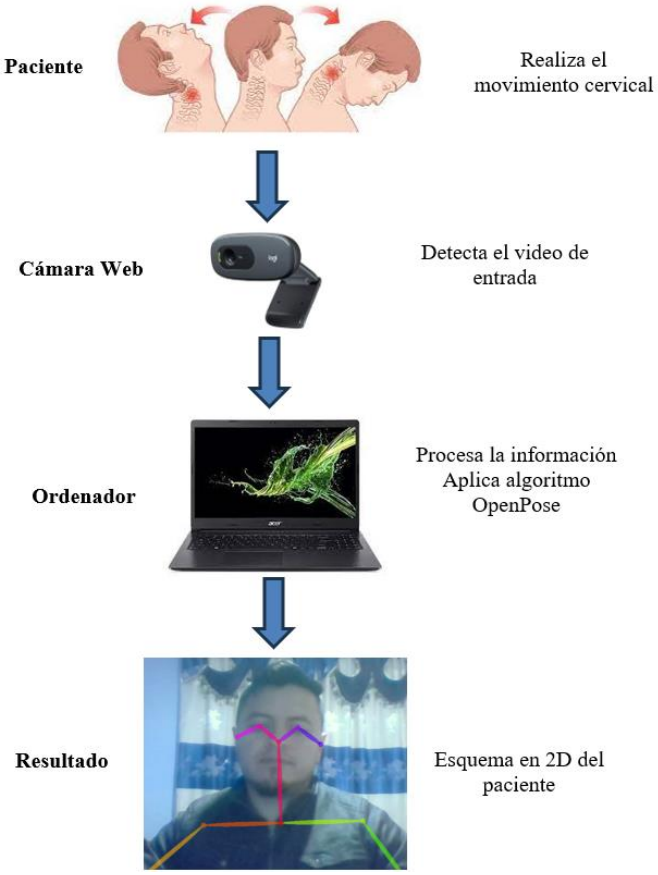
La Figura 13 ilustra la arquitectura para el sistema de medición de ángulos articulares durante los distintos movimientos cervicales. Este sistema se basa en el uso de una cámara web, la cual es responsable de la captura de datos mediante la detección de video del paciente que está siendo evaluado.

La cámara web está conectada al CPU a través de un puerto USB, permitiendo la transmisión en tiempo real de las imágenes adquiridas. A partir de estos datos, el sistema de estimación de posición humana OpenPose procesa la información y genera un esquema en 2D del paciente. Este esquema facilita la identificación y análisis de los ángulos articulares en los movimientos cervicales, proporcionando datos precisos para la evaluación.

Gracias a esta integración de hardware y software, el sistema permite una medición más eficiente y objetiva de los movimientos cervicales, lo que resulta útil para el diagnóstico, monitoreo y rehabilitación de los pacientes.

Figura 13

Arquitectura del sistema



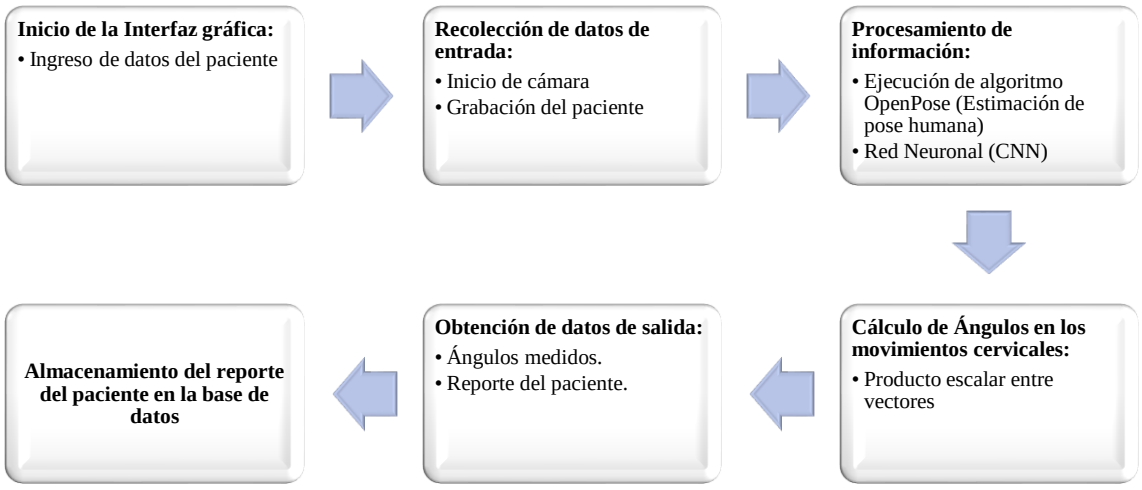
Fuente: Autoría

3.7.2. Diagrama de Bloques

El diagrama de bloques es una representación gráfica de alto nivel que muestra la estructura y el flujo de información dentro del sistema. Se compone de bloques que representan los principales componentes del sistema (como hardware, software y bases de datos) y líneas o flechas que indican la comunicación e interacción entre ellos. La Figura 14 muestra el diagrama del sistema compuesto por 6 bloques encargados de garantizar el funcionamiento del sistema.

Figura 14

Diagrama de bloques del sistema



Fuente: Autoría

3.7.2.1.Primer Bloque: Inicio de Interfaz Gráfica

Como etapa inicial del sistema se encuentra el inicio de la interfaz gráfica, donde se podrá usar una de las 4 opciones, tal como se muestra en la Figura 15. **Guardar:** al seleccionar esta opción el sistema guarda la información ingresada (datos del paciente). **Editar:** en la segunda opción se puede editar los datos de un paciente para luego de usar el sistema guardar las medidas encontradas en su perfil. **Diagnosticar:** permite realizar las pruebas de medición de ángulo en los movimientos cervicales del paciente. **Eliminar:** como su nombre lo indica, esta opcion permite eliminar registros que ya no se desee guardar en la base de datos.

Como última opción se encuentra una tabla donde se observa se puede revisar cada perfil previamente ingresado, esto en caso de tener que realizar nuevas consultas o editar algún parámetro de su perfil.

Figura 15

Inicio interfaz gráfica

Fuente: Autoría

3.7.2.2.Segundo Bloque: Recolección de datos de entrada

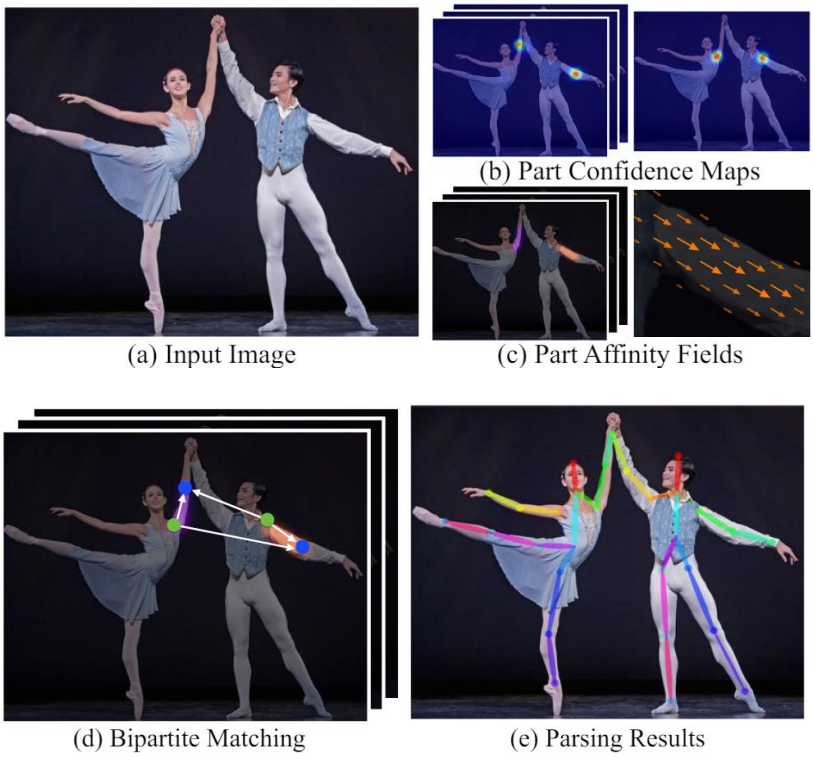
Para obtener información de calidad, es fundamental que las imágenes capturadas cuenten con una adecuada luminosidad, estén libres de obstrucciones e imperfecciones, y mantengan una escala constante. La cámara web utilizada proporcionará una captura en formato RGB. Para garantizar que el sistema preserve estas características de calidad, es crucial que la cámara convierta las señales luminosas en señales analógicas de manera eficiente.

Se ha seleccionado la cámara Logitech C270 debido a su enfoque fijo y su compatibilidad con resolución de 720p. La conexión de la Logitech C270 se realiza a través de un cable USB de 15 cm, que se conecta directamente al puerto USB del ordenador encargado de procesar el video.

3.7.2.3.Tercer Bloque: Procesamiento de la Información

Figura 16

Procesamiento de la información de OpenPose



Fuente: (Cao et al., 2017)

La Figura 16 muestra cómo a manera de resumen: (a) OpenPose toma la imagen completa como entrada para que una (b) CNN prediga conjuntamente mapas de confianza para la detección de partes del cuerpo y (c) PAFs para la asociación de partes. El paso de análisis (d) realiza un conjunto de coincidencias bipartitas para asociar candidatas de partes del cuerpo. (e) Finalmente los ensambla en poses de cuerpo completo para todas las personas de la imagen.

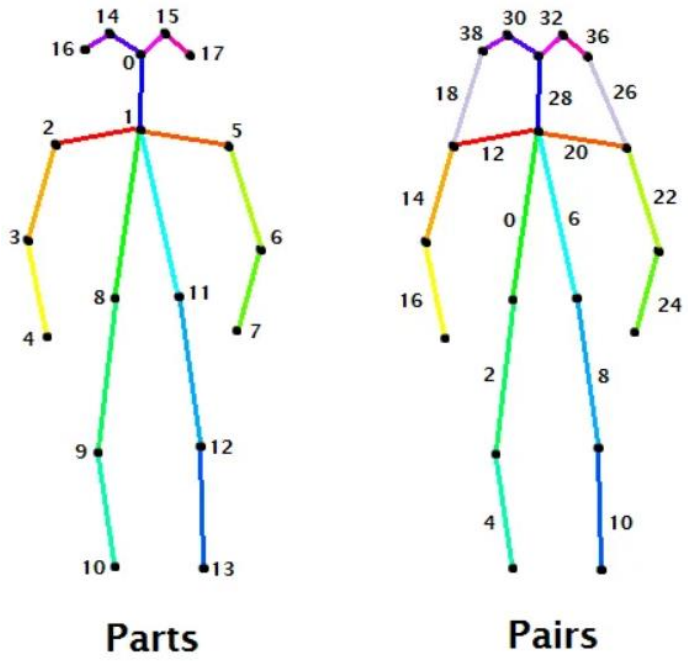
OpenPose reúne tres conjuntos de modelos entrenados: uno para la estimación de la postura del cuerpo, otro para las manos y el último para las caras. Cada conjunto tiene varios modelos dependiendo del conjunto de datos que se encuentren en (COCO Common Objects in Context) o MPII (Conjunto de datos de estimación articulada).

Una **parte** del cuerpo es un elemento del cuerpo, como el cuello, el hombro izquierdo o la muñeca derecha. Un **par** es la unión de dos partes. Una conexión entre partes se podría llamar una extremidad, pero la conexión entre la nariz y el ojo izquierdo definitivamente no es una extremidad (Cao et al., 2017).

Los esqueletos que se observan en la Figura17 muestran los índices de partes y pares en el conjunto de datos COCO. Para el conjunto de datos MPII, estos esqueletos varían ligeramente: hay una parte más del cuerpo correspondiente a los abdominales inferiores.

Figura 17

Partes y pares conjunto COCO



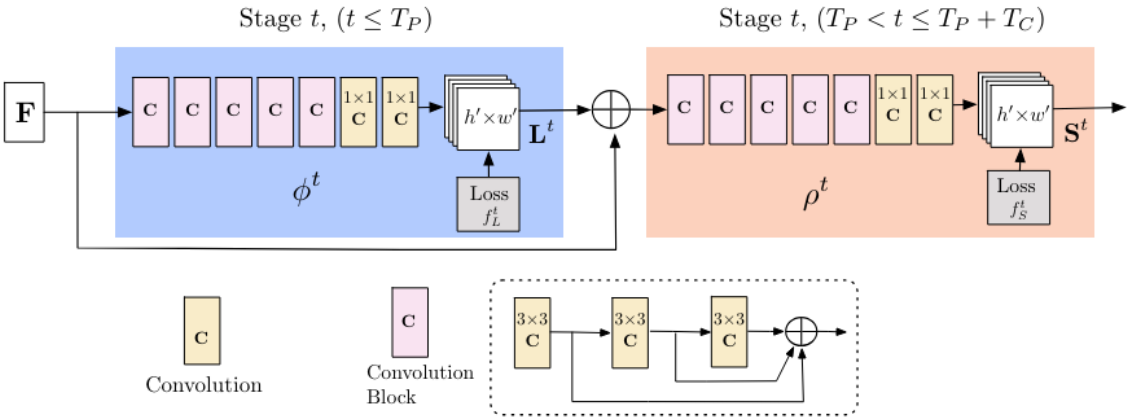
Fuente: (Cao et al., 2017)

Para realizar este proceso OpenPose hace uso de la siguiente arquitectura de red:

- **Arquitectura de la CNN multi etapa**

Figura 18

Arquitectura de la CNN de OpenPose



Fuente: (Cao et al., 2017)

En la Figura 18 se observa que el primer conjunto de etapas predice PAF L^t, mientras que el último conjunto predice mapas de confianza S^t. Las predicciones de cada etapa y sus características de imagen correspondientes se concatenan para cada etapa posterior. Las convoluciones de tamaño de núcleo 7 del enfoque original se reemplazan con 3 capas de convoluciones de núcleo 3 que se concatenan en su extremo(Cao et al., 2017).

En el enfoque original, la arquitectura de red incluía varias capas convolucionales de 7x7. En nuestro modelo actual, el campo receptivo se conserva mientras se reduce el cálculo, reemplazando cada núcleo convolucional de 7x7 por 3 núcleos consecutivos de 3x3. Mientras que el número de operaciones para el primero es $2 * 7^2 - 1 = 97$, para el segundo es solo 51. Además, la salida de cada uno de los 3 núcleos convolucionales se concatena. El número de capas no lineales se triplica y la red puede mantener características tanto de nivel inferior como superior (Cao et al., 2017).

Etapa 1: En la primera etapa se calcula los campos de afinidad de piezas (PAF) L1, a partir de los mapas de características de la red base F. Sea $\emptyset^1$ la CNN en la etapa 1.

$$L^1 = \phi^1(F)$$

Etapa t a etapa Tp: refina las predicciones de los PAF de la etapa anterior utilizando los mapas de características F y los PAF anteriores Lt-1. Sea $\emptyset^t$ la CNN en la etapa t.

$$L^t = \phi^t(F, L^{t-1}), \forall 2 \leq t \leq T_P$$

Después de las iteraciones de Tp, el proceso se repite para la detección de mapas de confianza, comenzando en la predicción PAF más actualizada. Sea ρ^t la CNN en la etapa t. El proceso se repite para la iteración Tc (Cao et al., 2017).

$$S^{T_P} = \rho^t(F, L^{T_P}), \forall t = T_P$$

$$S^t = \rho^t(F, L^{T_P}, S^{t-1}), \forall T_P < t \leq T_P + T_C$$

Los S y L finales son los mapas de confianza y los campos de afinidad de partes (PAF) que serán procesados posteriormente por el algoritmo codicioso.

Para guiar a la red a predecir de forma iterativa los PAF de partes del cuerpo en la primera rama y los mapas de confianza en la segunda rama, aplicamos una función de pérdida al final de cada etapa. Usamos una pérdida L2 entre las predicciones estimadas y los mapas y campos de verdad fundamental. La máscara se utiliza para no penalizar las predicciones positivas verdaderas durante el entrenamiento (Cao et al., 2017). La supervisión intermedia en cada etapa aborda el problema del gradiente que desaparece reponiendo el gradiente periódicamente. El objetivo general es:

$$f = \sum_{t=1}^{T_P} f_{\mathbf{L}}^t + \sum_{t=T_P+1}^{T_P+T_C} f_{\mathbf{S}}^t. \qquad (1)$$

Para evaluar f_s en la Ecuación 1 durante el entrenamiento, generamos los mapas de confianza de la verdad fundamental S* de los puntos clave 2D anotados. Cada mapa de confianza es una representación 2D de la creencia de que una parte del cuerpo en particular puede ubicarse en cualquier píxel determinado. Idealmente, si aparece una sola persona en la imagen, debería existir un único pico en cada mapa de confianza si la parte correspondiente es visible; Si hay varias personas en la imagen, debe haber un pico correspondiente a cada parte visible j para cada persona k (Cao et al., 2017).

$$\mathbf{S}_{j,k}^*(\mathbf{p}) = \exp\left(-\frac{||\mathbf{p} - \mathbf{x}_{j,k}||_2^2}{\sigma^2}\right)$$

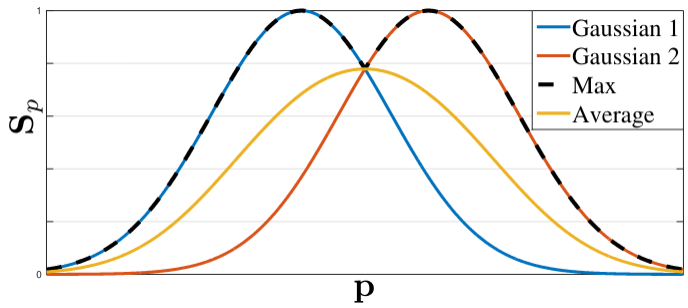
donde σ controla la propagación del pico. El mapa de confianza de la verdad fundamental predicho por la red es una agregación de los mapas de confianza individuales a través de un operador máximo,

$$\mathbf{S}_j^*(\mathbf{p}) = \max_k \mathbf{S}_{j,k}^*(\mathbf{p}).$$

Tomamos el máximo de los mapas de confianza en lugar del promedio para que la precisión de los picos cercanos siga siendo distinta, como se ilustra en la Figura 19. En el momento de la prueba, predecimos mapas de confianza y obtenemos partes del cuerpo candidatas realizando una supresión no máxima tal como se muestra en la Figura 20 (Cao et al., 2017).

Figura 19

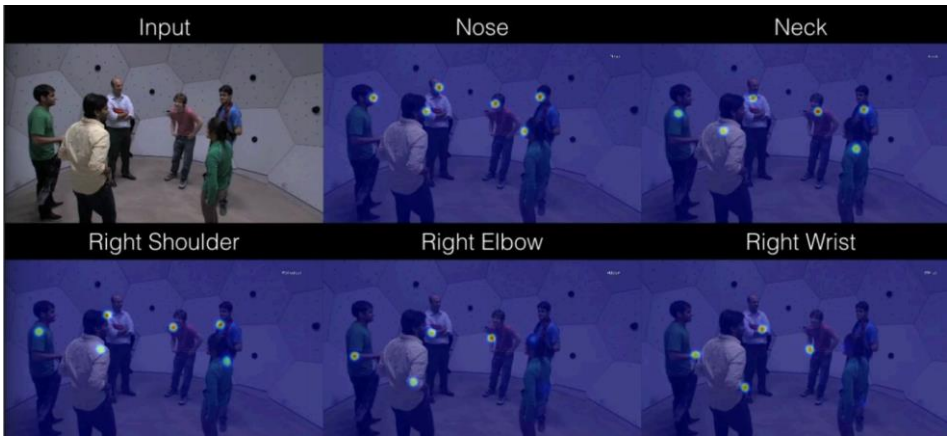
Mapas de confianza



Fuente: (Cao et al., 2017)

Figura 20

Obtención de partes del cuerpo



Fuente: (Cao et al., 2017)

En la Figura 21 se muestra las estrategias de asociación de piezas. (a) Los candidatos a detección de partes del cuerpo (puntos rojos y azules) para dos tipos de partes del cuerpo y todos los candidatos a conexión (líneas grises). (b) Los resultados de la conexión utilizando la representación del punto medio (puntos amarillos): conexiones correctas (líneas negras) y conexiones incorrectas (líneas verdes) que también satisfacen la restricción de incidencia. (c) Los resultados utilizando PAF (flechas amarillas). Al codificar la posición y la orientación sobre el soporte de la extremidad, los PAF eliminan falsas asociaciones (Cao et al., 2017).

Figura 21

Estrategias de asociación de piezas

Fuente: (Cao et al., 2017)

Durante las pruebas, medimos la asociación entre las detecciones de piezas candidatas calculando la integral de línea sobre el PAF correspondiente a lo largo del segmento de línea que conecta las ubicaciones de las piezas candidatas. En otras palabras, medimos la alineación del PAF previsto con la extremidad candidata que se formaría conectando las partes del cuerpo detectadas. Específicamente, para dos ubicaciones de piezas candidatas d_{j1} y d_{j2}, tomamos una muestra del campo de afinidad de piezas predicho, Lc, a lo largo del segmento de línea para medir la confianza en su asociación (Cao et al., 2017):

$$E = \int_{u=0}^{u=1} \mathbf{L}_c\left(\mathbf{p}(u)\right) \cdot \frac{\mathbf{d}_{j_2} - \mathbf{d}_{j_1}}{||\mathbf{d}_{j_2} - \mathbf{d}_{j_1}||_2} du$$

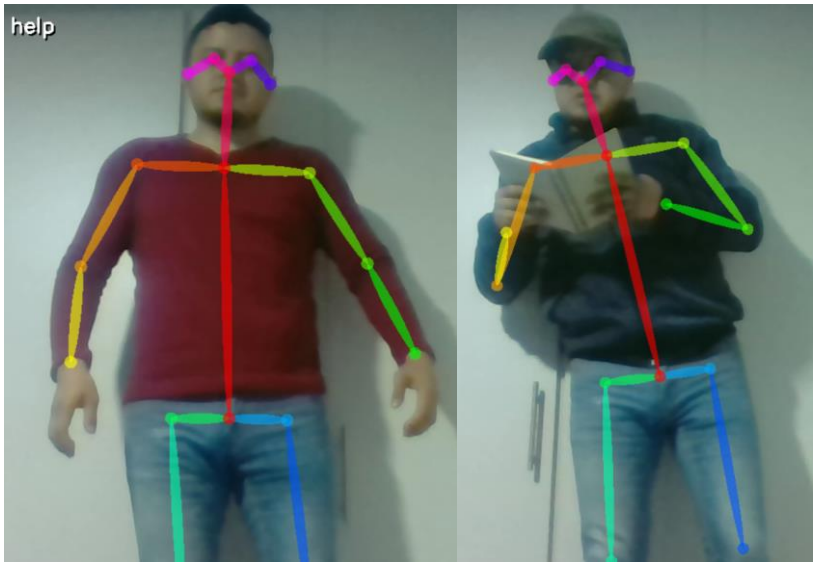
Donde p(u) interpola la posición de las dos partes del cuerpo d_{j1} y d_{j2},

$$\mathbf{p}(u) = (1-u)\mathbf{d}_{j_1} + u\mathbf{d}_{j_2}$$

Finalmente, lo que se obtiene es una colección de pares (extremidades) del cuerpo humano, donde cada par es un conjunto de dos partes, cada parte contiene su índice, sus coordenadas relativas y su puntaje como indica la Figura 22.

Figura 22

Obtención de pares (extremidades) del cuerpo humano



Fuente: Autoría

3.7.2.4.Cuarto Bloque: Cálculo de Ángulos en movimientos cervicales

En este apartado, se realiza un análisis del cálculo de los ángulos articulares utilizando el algoritmo de estimación de pose humana. El objetivo de este bloque es aprovechar el código abierto proporcionado por OpenPose, el cual está disponible bajo licencia para desarrollos en investigaciones académicas.

Es importante resaltar la instalación de los drivers actualizados para la tarjeta de video, ya que estos son esenciales para la correcta instalación de los drivers CUDA. Estos drivers son necesarios para habilitar los procesos en la GPU, lo que permitirá un rendimiento óptimo en el uso de TensorFlow, OpenCV y Python. Cabe señalar que el

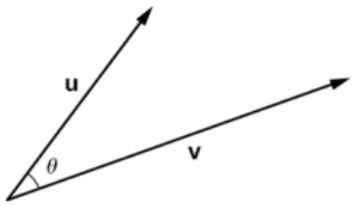
procesamiento y uso de OpenPose se ejecutan principalmente en la GPU, minimizando el uso del CPU.

Para el cálculo de los ángulos cervicales es necesario conocer los índices de partes (puntos clave) del conjunto COCO que se va a utilizar, esto se puede visualizar en la Figura 17. Para el cálculo de los ángulos de inclinación lateral derecha e izquierda se necesitan los siguientes puntos clave: {0, 1, 2, 5} correspondientes a los puntos de {nariz, cuello, hombro derecho, hombro izquierdo} respectivamente.

Una vez conocidos los puntos clave se procede a realizar un producto escalar entre los vectores que se forman al unir dichos puntos. El producto escalar entre dos vectores está definido como el producto de la magnitud (módulo) de cada vector por el coseno del ángulo que existe entre ellos, ver Figura 23.

Figura 23

Producto escalar entre dos vectores



$$u \,.\, v = \|u\|\|v\| \cos\theta$$

Fuente: autoría

De la ecuación anterior se puede despejar θ que vendría a ser el ángulo que se necesitaba encontrar, quedando una nueva ecuación de la forma:

$$\theta = cos^{-1}\left(\frac{u \,.\, v}{\|u\|\|v\|}\right)$$

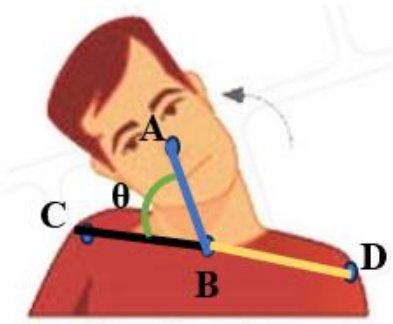
Integrando la fórmula obtenida al sistema que se va a implementar, en primer lugar, se asigna una letra a cada uno de los puntos clave, luego se encuentra los vectores

que se forman al unir dichos puntos y finalmente se calcula el ángulo que se forma entre dichos vectores, tal como se muestra en la figura 24.

A=0 ; B=1 ; C=2 ; D=5

Figura 24

Asignación de letras a puntos clave y formación de vectores



Fuente: Autoría

El ángulo θ se encuentra utilizando la ecuación

$$\theta = cos^{-1}\left(\frac{BA\,.BC}{\|BA\|\|BC\|}\right)$$

Donde,

$$BA = A - B \quad ; \quad BC = C - B \quad ; \quad BD = D - B$$

$$\|BA\| = \sqrt{(BA_x)^2 + \left(BA_y\right)^2}$$

$$\|BC\| = \sqrt{(BC_x)^2 + \left(BC_y\right)^2}$$

Lo antes mencionado para calcular el ángulo de inclinación derecha o izquierda programado en Phyton quedaría tal como se muestra en la Figura 25.

Figura 25

Cálculo de ángulo formado por 2 vectores en Python

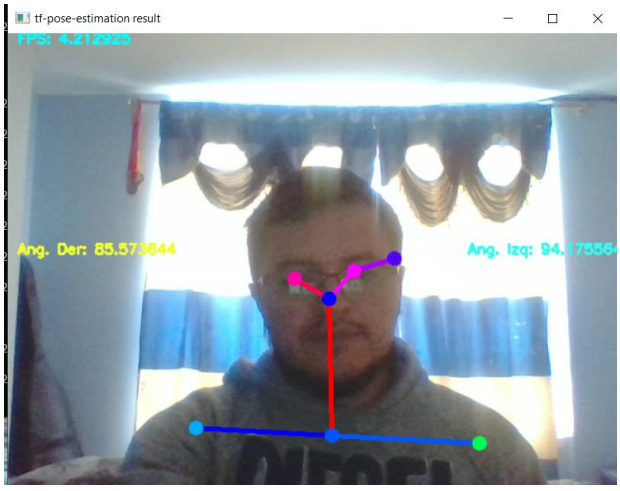
```
if(centers[0] != "" and centers[1] != "" and centers[2] != "" and centers[5] != ""):

    A = centers[0]
    B = centers[1]
    C = centers[2]
    D = centers[5]

    # VECTORES

    BA = np.array([A[0] - B[0], A[1] - B[1]])
    BC = np.array([C[0] - B[0], C[1] - B[1]])
    BD = np.array([D[0] - B[0], D[1] - B[1]])

    # MODULOS O DISTANCIA ENTRE PUNTOS

    MOD_BA = math.sqrt( (A[0] - B[0])**2 + (A[1] - B[1])**2)
    MOD_BC = math.sqrt( (C[0] - B[0])**2 + (C[1] - B[1])**2)
    MOD_BD = math.sqrt( (D[0] - B[0])**2 + (D[1] - B[1])**2)

    # PRODUCTO ESCALAR ENTRE VECTORES V1@V2
    # FORMULA -> V1.V2 = |V1|.|V2|*cos(theta)

    P_ESCALAR_BA_BC = BA@BC
    P_ESCALAR_BA_BD = BA@BD

    # CALCULO DEL ANGULO THETA 1 -> LADO DERECHO
    # CALCULO DEL ANGULO THETA 2 -> LADO IZQUIERDO

    COS_THETHA1 = (P_ESCALAR_BA_BC)/(MOD_BA*MOD_BC)
    COS_THETHA2 = (P_ESCALAR_BA_BD)/(MOD_BA*MOD_BD)

    theta1 = math.degrees(math.acos(COS_THETHA1))
    theta2 = math.degrees(math.acos(COS_THETHA2))
```

Fuente: Autoría

Al ejecutar el script de la Figura 25, se obtienen los ángulos de inclinación lateral derecha e izquierda tal como se muestra en la Figura 26.

Figura 26

Ángulos formados entre la nariz, cuello y hombros



Fuente: Autoría

3.7.2.5.Quinto Bloque: Obtención de datos de Salida

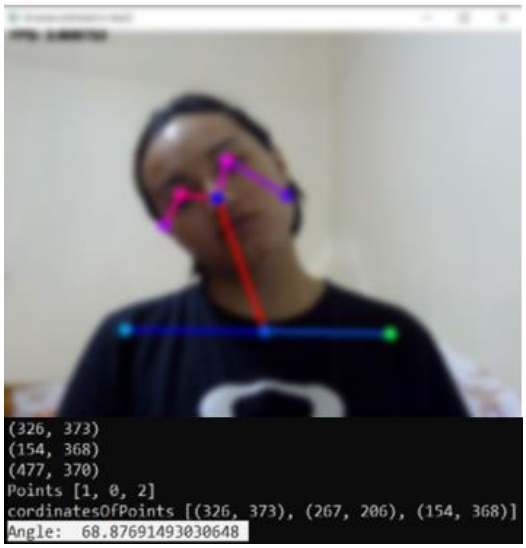
En este bloque se procede a obtener los resultados de las mediciones realizadas en cada tipo de movimiento cervical par luego poder comparar con los datos medidos por el profesional correspondiente de manera manual.

3.7.2.5.1. Inclinación lateral derecha

Para obtener el ángulo medido en este movimiento, el paciente se debe colocar sentado viendo de frente a la camara y mover su cabeza lo maximo posible hacia el lado derecho sin inclinar su torso, tal como se muestra en la Figura 27.

Figura 27

Obtención de ángulo en Inclinación lateral derecha.



Fuente: Autoría

3.7.2.5.2. Inclinación lateral izquierda

Para obtener el ángulo medido en este movimiento, el paciente se debe colocar sentado viendo de frente a la camara y mover su cabeza lo maximo posible hacia el lado izquierdo sin inclinar su torso, tal como se muestra en la Figura 28.

Figura 28

Obtención de ángulo en Inclinación lateral izquierda.

Fuente: Autoría

3.7.2.5.3. Movimiento de extensión

Para obtener el ángulo medido en este movimiento, el paciente se debe colocar sentado y de perfil a la camara, luego debe mover su cabeza lo maximo posible hacia arriba sin inclinar su torso, tal como se muestra en la Figura 29.

Figura 29

Obtención de ángulo en movimiento de extensión.

Fuente: Autoría

3.7.2.5.4. Movimiento de flexión

Para obtener el ángulo medido en este movimiento, el paciente se debe colocar sentado y de perfil a la camara, luego debe mover su cabeza lo maximo posible hacia arriba sin inclinar su torso, tal como se muestra en la Figura 30.

Figura 30

Obtención de ángulo en movimiento de Flexión.

Fuente: Autoría

3.7.2.5.5. Movimiento de rotación derecha

Para obtener el ángulo medido en este movimiento, el paciente se debe colocar sentado y de perfil a la cámara, luego debe mover su cabeza lo maximo posible hacia la derecha sin inclinar su torso, tal como se muestra en la Figura 31.

Figura 31

Obtención de ángulo en movimiento de rotación derecha.

Fuente: Autoría

3.7.2.5.6. Movimiento de rotación izquierda

Para obtener el ángulo medido en este movimiento, el paciente se debe colocar sentado y de perfil a la camara, luego debe mover su cabeza lo maximo posible hacia la izquierda sin inclinar su torso, tal como se muestra en la Figura 32.

Figura 32

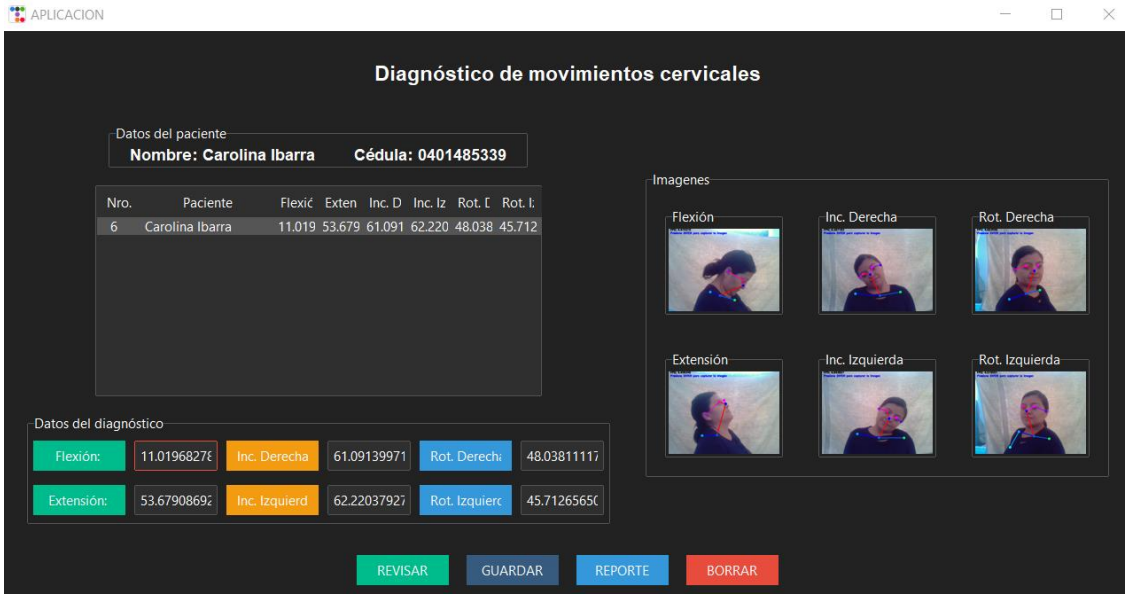
Obtención de ángulo en movimiento de rotación izquierda

Fuente: Autoría

Una vez calculados los ángulos de cada uno de los movimientos cervicales en el paciente se procede a guardar los mismo dando click en el botón Guardar en la interfaz gráfica, y de ser necesario, se puede realizar una nueva toma de mediciones en el mismo paciente, tal como se muestra en la figura 33. Este proceso se debe repetir una vez se haya realizado la terapia de rehabilitación en el paciente para comparar si habido o no mejoras.

Figura 33

Diagnóstico medido en el paciente



Fuente: Autoría

3.7.2.6.Sexto Bloque: Almacenamiento en la base de datos4

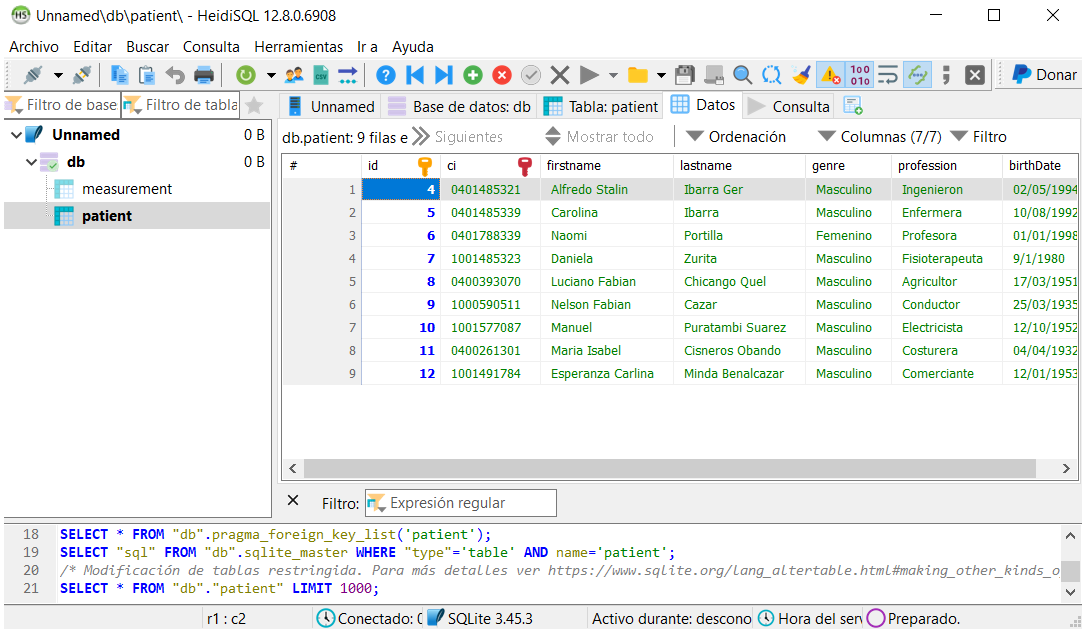
Para almacenar los datos medidos en cada paciente, se ha utilizado el módulo Python SQLite3, el cual se utiliza para integrar la base de datos SQLite con Python. Es una API DBI 2.0 de Python estandarizada y proporciona una interfaz sencilla y fácil de usar para interactuar con bases de datos SQLite. SQLite es un motor de base de datos

SQL de dominio público, autónomo, altamente confiable, integrado y con todas las funciones. Es el motor de base de datos más utilizado en la red mundial (Juan Sánchez Hernández, 2023).

En la Figura 15 se ingresan los datos del paciente, los cuales deben ser almacenados en una tabla de la base de datos denominada “patient” tal como se muestra en la Figura 34. La base de datos para este proyecto se llama “db”.

Figura 34

Tabla de datos “patient”

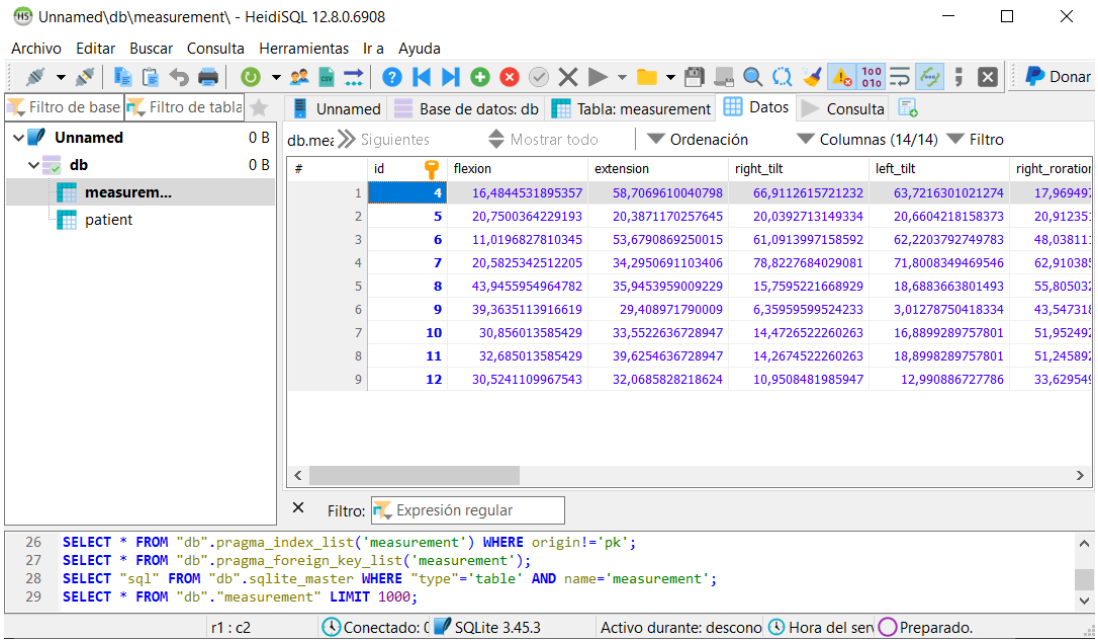


Fuente: Autoría

Luego se realiza las mediciones de los distintos ángulos (ver Figura 33) y estos datos se guardan en la tabla “measurement” tal como se muestra en la Figura 35.

Figura 35

Tabla de datos "measurement"

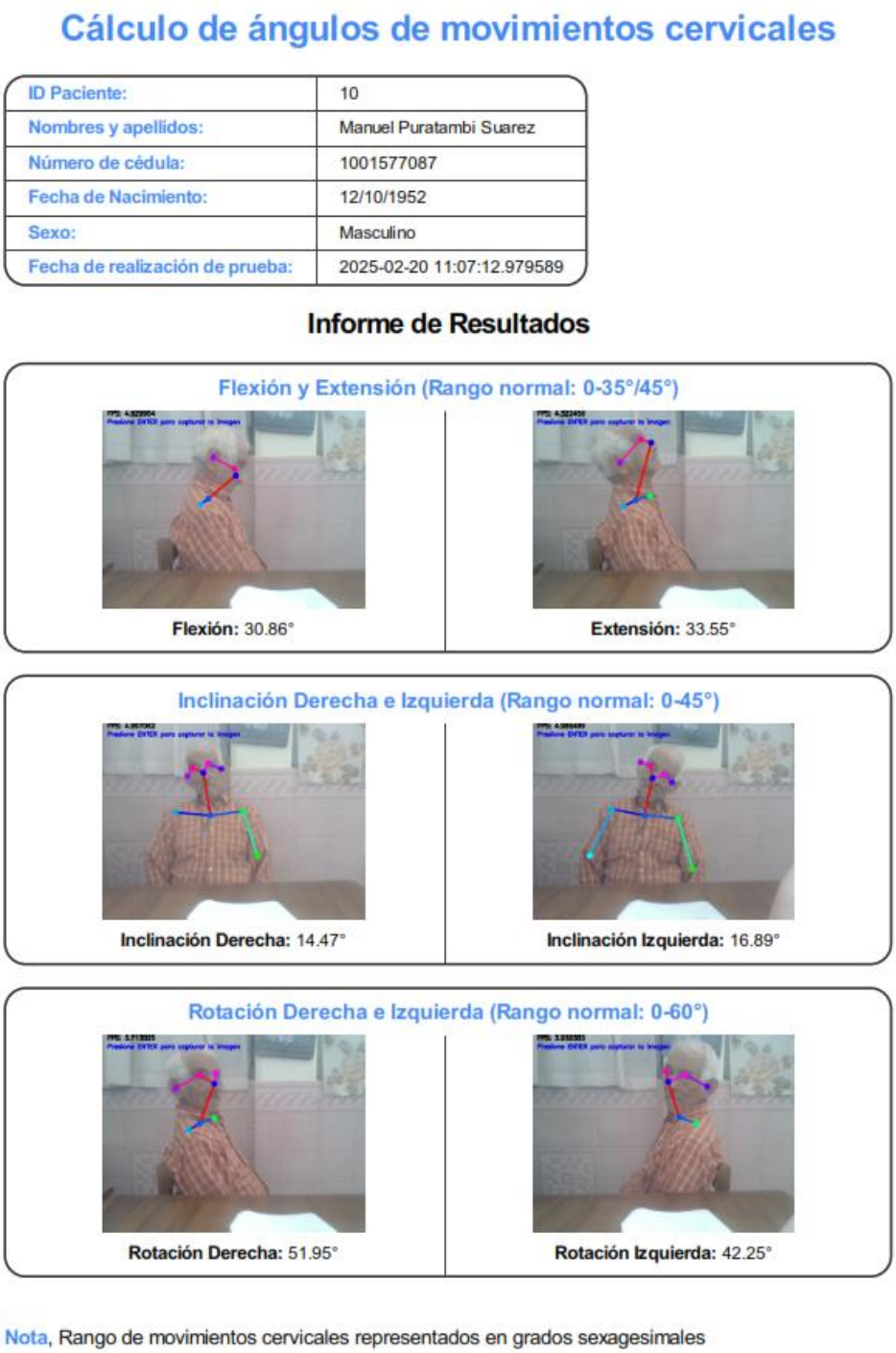


Fuente: Autoría

Finalmente, debido a la necesidad de obtener los diagnósticos de manera física, se implementó un botón denominado **"REPORTE"** (ver Figura 33). Este botón permite generar y exportar un formulario en formato PDF, que incluye los ángulos medidos para cada tipo de movimiento cervical, acompañados de las imágenes correspondientes. Esto facilita la verificación de los resultados, proporcionando una forma clara y confiable de comprobar la exactitud de los datos obtenidos. El formulario antes mencionado se observa en la Figura 36.

Figura 36

Formulario generado de un paciente



Cálculo de ángulos de movimientos cervicales

ID Paciente:	10
Nombres y apellidos:	Manuel Puratambi Suarez
Número de cédula:	1001577087
Fecha de Nacimiento:	12/10/1952
Sexo:	Masculino
Fecha de realización de prueba:	2025-02-20 11:07:12.979589

Informe de Resultados

Flexión y Extensión (Rango normal: 0-35°/45°)

Flexión: 30.86°

Extensión: 33.55°

Inclinación Derecha e Izquierda (Rango normal: 0-45°)

Inclinación Derecha: 14.47°

Inclinación Izquierda: 16.89°

Rotación Derecha e Izquierda (Rango normal: 0-60°)

Rotación Derecha: 51.95°

Rotación Izquierda: 42.25°

Nota, Rango de movimientos cervicales representados en grados sexagesimales

Fuente: Autoría

4. Pruebas de Funcionamiento y Resultados

Este capítulo presenta los resultados obtenidos tras la ejecución de las pruebas de funcionamiento del sistema en el Centro Gerontológico Residencial León Ruales.

Además, tras obtener los datos de las pruebas de funcionamiento del sistema, se realiza una comparación con las mediciones del profesional de salud encargado. Esto permite evaluar cómo se superaron las restricciones a lo largo del desarrollo del proyecto y determinar si el sistema cumple con los objetivos iniciales propuestos.

4.1.Pruebas de funcionamiento del Sistema

Una vez terminada la etapa del diseño del sistema se procede a realizar las pruebas que validen todo lo propuesto en el proyecto, la finalidad de estas pruebas es verificar que se tenga datos con coherencia objetiva para comparar la medición del sistema versus la medición manual.

Para comparar datos medidos manualmente con datos obtenidos digitalmente, primero debemos asegurar que las dos mediciones se realicen bajo las mismas condiciones y utilizando el mismo sistema de unidades, que para el caso son los grados sexagesimales.

Las mediciones manuales se realizarán utilizando el método tradicional del goniómetro, mientras que las mediciones digitales se realizarán con el sistema implementado usando una cámara con OpenPose.

Para la prueba de funcionamiento del sistema se cuenta con el criterio de un experto, la MsC. Daniela Zurita. La evaluación del experto permite realizar una comparación entre la evaluación del sistema y la de la medición manual.

4.1.1. Evaluación de movimiento de Flexión – Extensión

Es importante que las mediciones tanto manuales como digitales se realicen en el mismo período de tiempo y bajo las mismas circunstancias, ya que un cambio en estos parámetros podría causar variaciones en las posturas de pacientes, lo cual alteraría los datos obtenidos por la aplicación.

Una vez realizadas la medición manual en el paciente (ver Figura 37a y 38a) con ayuda del goniómetro, se procedió a realizar la toma de datos digital (ver Figura 37b y 38b) por medio de la aplicación diseñada. Llegando a obtener los resultados plasmados en la tabla 14 y 15.

Figura 37

Medición manual (a) y digital (b) de ángulo en movimiento de Extensión.

(a) (b)

Fuente: Autoría

Figura 38

Medición manual (a) y digital (b) de ángulo en movimiento de Flexión.

(a) (b)

Fuente: Autoría

Tabla 14

Datos medidos manual y digitalmente en movimientos de Flexión.

Paciente	Flexión		Diferencia entre medidas A y B	Validación
	Aplicación (A)	Goniómetro (B)		
1	43,94	41	-2,94	SI CUMPLE
2	39,36	37	-2,36	SI CUMPLE
3	30,85	28	-2,85	SI CUMPLE
4	32,68	34	1,32	SI CUMPLE
5	30,52	32	1,48	SI CUMPLE

Fuente: Autoría

Tabla 15

Datos medidos manual y digitalmente en movimientos de Extensión.

Paciente	Extensión		Diferencia entre medidas A y B	Validación
	Aplicación (A)	Goniómetro (B)		
1	35,94	33	-2,94	SI CUMPLE
2	29,4	30	0,6	SI CUMPLE
3	33,55	32	-1,55	SI CUMPLE
4	39,62	38	-1,62	SI CUMPLE
5	33,06	36	2,94	SI CUMPLE

Fuente: Autoría

Como se puede apreciar en la Tabla 14 y en la Tabla 15, los datos obtenidos con el sistema no presentan una diferencia superior o inferior a ±5 grados sexagesimales en comparación con las mediciones manuales. Esto permite concluir que el sistema diseñado “SI CUMPLE” con el estándar médico establecido por los profesionales de salud, sin mostrar diferencias significativas en relación con las mediciones manuales realizadas.

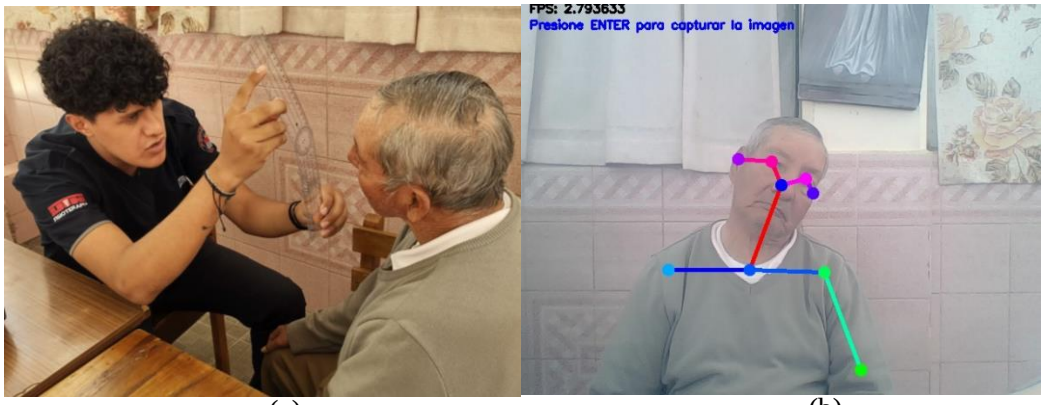
4.1.2. Evaluación de movimiento de Inclinación Lateral Derecha e Izquierda

Se realiza la medición manual en el paciente (ver Figura 39a y 40a) con ayuda del goniómetro por parte del profesional de fisioterapia, a continuación, se captura los datos digitales (ver Figura 39b y 40b) por medio de la aplicación diseñada. Llegando a obtener los resultados plasmados en la tabla 16 y 17.

Para garantizar la precisión de los datos obtenidos por la aplicación, las mediciones manuales y digitales deben realizarse simultáneamente y bajo condiciones idénticas. Cualquier variación en estos factores podría afectar la postura de los pacientes, generando discrepancias en los resultados.

Figura 39

Medición manual (a) y digital (b) de ángulo en movimiento de Inclinación izquierda

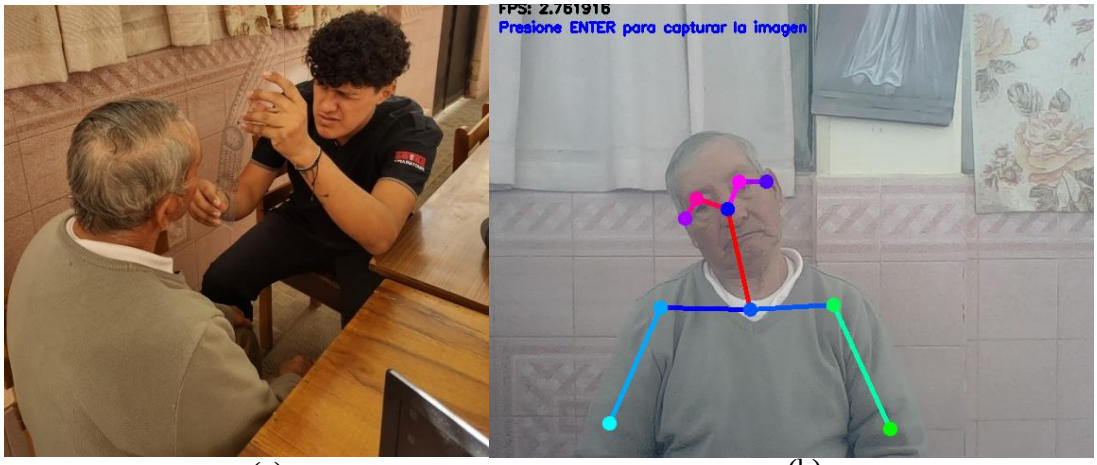


(a) (b)

Fuente: Autoría

Figura 40

Medición manual (a) y digital (b) de ángulo en movimiento de Inclinación derecha



(a) (b)

Fuente: Autoría

Tabla 16

Datos medidos manual y digitalmente del movimiento de inclinación derecha.

Paciente	Inclinación Derecha		Diferencia entre medidas A y B	Validación
	Aplicación (A)	Goniómetro (B)		
1	15,75	14	-1,75	SI CUMPLE
2	6,36	5	-1,36	SI CUMPLE
3	14,47	12	-2,47	SI CUMPLE
4	14,26	11	-3,26	SI CUMPLE
5	10,95	8	-2,95	SI CUMPLE

Fuente: Autoría

Tabla 17

Datos medidos manual y digitalmente del movimiento de inclinación izquierda.

Paciente	Inclinación Derecha		Diferencia entre medidas A y B	Validación
	Aplicación (A)	Goniómetro (B)		
1	18,69	19	0,31	SI CUMPLE
2	3,01	3	-0,01	SI CUMPLE
3	16,89	18	1,11	SI CUMPLE
4	18,9	20	1,1	SI CUMPLE
5	13	12	-1	SI CUMPLE

Fuente: Autoría

Como se puede apreciar en la Tabla 16 y en la Tabla 17, los datos obtenidos con el sistema no presentan una diferencia superior o inferior a ±5 grados sexagesimales en comparación con las mediciones manuales. Esto permite concluir que el sistema diseñado “SI CUMPLE” con el estándar médico establecido por los profesionales de salud, sin mostrar diferencias significativas en relación con las mediciones manuales realizadas.

.

4.1.3. Evaluación de movimiento de Rotación Derecha e Izquierda

Se realiza la medición manual en el paciente (ver Figura 41a y 42a) con ayuda del goniómetro por parte del profesional de fisioterapia, a continuación, se captura los datos digitales (ver Figura 41b y 42b) por medio de la aplicación diseñada. Llegando a obtener los resultados plasmados en la tabla 18 y 19.

Figura 41

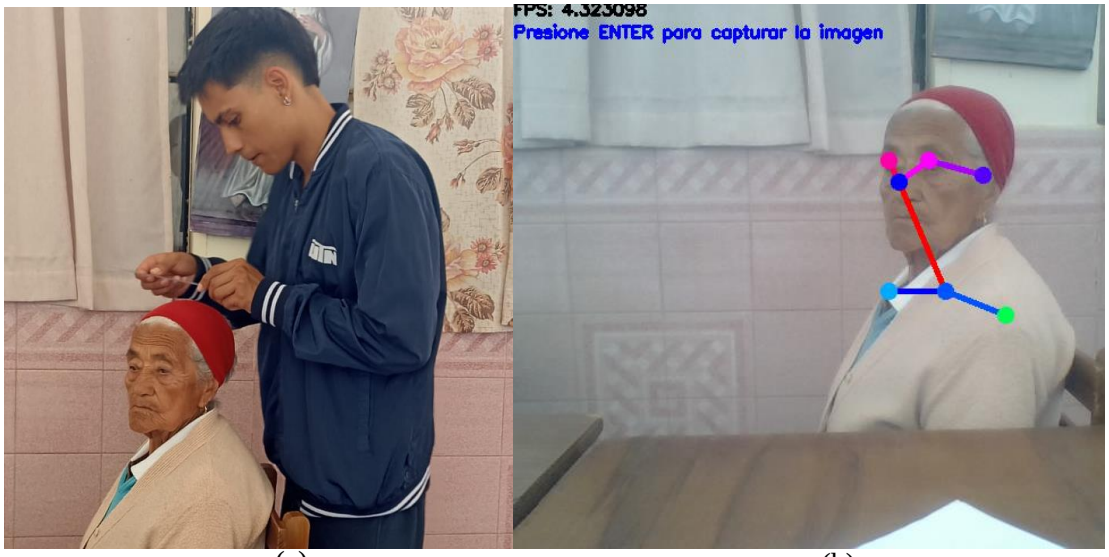
Medición manual (a) y digital (b) de ángulo en movimiento de Rotación derecha.

(a) (b)

Fuente: Autoría

Figura 42

Medición manual (a) y digital (b) de ángulo en movimiento de Rotación izquierda.



(a) (b)

Fuente: Autoría

Se repite el proceso para cada uno de los pacientes y se anotan los resultados obtenidos para luego poder comparar entre sí, tal como se observa en la Tabla 18 y en la Tabla 19.

Tabla 18.

Datos medidos manual y digitalmente del movimiento de rotación derecha.

Paciente	Rotación Derecha		Diferencia entre medidas A y B	Validación
	Aplicación (A)	Goniómetro (B)		
1	55,8	60	4,2	SI CUMPLE
2	43,54	39	-4,54	SI CUMPLE
3	51,95	47	-4,95	SI CUMPLE
4	54,24	50	-4,24	SI CUMPLE
5	33,63	29	-4,63	SI CUMPLE

Fuente: Autoría

Tabla 19

Datos medidos manual y digitalmente del movimiento de rotación izquierda.

Paciente	Rotación Izquierda		Diferencia entre medidas A y B	Validación
	Aplicación (A)	Goniómetro (B)		
1	46,87	51	4,13	SI CUMPLE
2	39,24	44	4,76	SI CUMPLE
3	42,25	47	-4,75	SI CUMPLE
4	60,52	65	4,48	SI CUMPLE
5	48,07	44	-4,07	SI CUMPLE

Fuente: Autoría

Como se puede apreciar en la Tabla 18 y en la Tabla 19, los datos obtenidos con el sistema no presentan una diferencia superior o inferior a ±5 grados sexagesimales en comparación con las mediciones manuales. Esto permite concluir que el sistema diseñado "SI CUMPLE" con el estándar médico establecido por los profesionales de salud, sin mostrar diferencias significativas en relación con las mediciones manuales realizadas.

Los valores de la columna "Diferencia entre medidas A y B" en las Tablas 14, 15, 16, 17, 18 y 19 no representan únicamente cifras positivas o negativas, sino que indican la dirección de la variación en la medición de los ángulos. En este sentido, el signo '+' denota una percepción más hacia la derecha o hacia arriba, mientras que el signo '-' señala una percepción más hacia la izquierda o hacia abajo.

5. Conclusiones

Se logró diseñar un sistema capaz de recibir imágenes, procesarlas y estimar la postura humana para obtener con precisión los ángulos en los movimientos cervicales. Este desarrollo tiene como objetivo automatizar el proceso de rehabilitación física en adultos mayores, facilitando su recuperación, optimizando el seguimiento terapéutico y contribuyendo a la mejora de su movilidad y calidad de vida.

Con base en los datos obtenidos, se puede observar que los movimientos de rotación, tanto hacia la derecha como hacia la izquierda, presentan una diferencia mayor entre las mediciones manuales y digitales en comparación con los demás movimientos analizados, superando en todos los casos los cuatro grados de diferencia. Por otro lado, en los movimientos de flexión, extensión e inclinación, la diferencia entre medición manual y digital observada se mantiene por debajo de los cuatro grados sexagesimales.

Los datos obtenidos con el sistema muestran una variación no mayor a ±5 grados sexagesimales en comparación con las mediciones manuales. Esto confirma que el sistema diseñado cumple con el estándar médico establecido por los profesionales de la salud, ya que no presenta diferencias significativas respecto al método tradicional. Tras su implementación y evaluación, se verificó que satisface los criterios de funcionalidad, precisión y rendimiento definidos previamente, demostrando su fiabilidad y adecuación para el propósito previsto.

La interfaz gráfica de usuario de su traducción del inglés Graphical User Interface (GUI) diseñada para el proyecto es intuitiva, funcional y visualmente atractiva, optimizando la interacción entre el usuario y el sistema. Su diseño sigue estándares comunes para garantizar una experiencia predecible y sencilla, eliminando elementos innecesarios que puedan generar distracción o confusión. Además, incluye un formulario

que muestra los datos del paciente registrado, junto con las mediciones de los ángulos obtenidas, facilitando el acceso y la interpretación de la información.

La estimación de la postura es una tarea de visión artificial cuyo objetivo es detectar la posición y la orientación de una persona o un objeto. Por lo general, esto se hace prediciendo la ubicación de puntos clave específicos, como las manos, la cabeza, los codos, entre otros., en el caso de la estimación de la postura humana. Encuentra aplicaciones en una amplia gama de campos, incluidos los juegos, la atención médica, la realidad aumentada y los deportes.

6. Recomendaciones

El sistema de detección de ángulos en movimientos cervicales opera mediante la captura de imágenes a través de una cámara web. Para garantizar un rendimiento óptimo, se recomienda realizar las pruebas o la toma de datos en un entorno con iluminación adecuada, lo que permitirá una mejor calidad en las imágenes y un correcto funcionamiento del sistema.

La cámara debe permanecer estable y fija, evitando cualquier movimiento, ya que todas las capturas de imágenes deben realizarse desde el mismo punto de referencia. Esto es especialmente importante al medir los ángulos de los movimientos de rotación, donde se requieren dos capturas de imagen para una evaluación precisa.

Los sistemas de detección de pose humana identifican las articulaciones del cuerpo como puntos de referencia clave. Se recomienda que los pacientes eviten usar ropa demasiado grande o holgada, ya que esto podría dificultar la detección precisa de sus articulaciones, afectando la capacidad del algoritmo para calcular los ángulos correspondientes de manera adecuada.

Es fundamental prestar especial atención a la organización de los cables y extensiones, asegurando que no representen un riesgo de tropiezo. Dado que los pacientes a tratar son adultos mayores, una instalación inadecuada del sistema podría generar accidentes, comprometiendo su seguridad y el entorno de trabajo.

El paciente debe estar correctamente posicionado frente a la cámara, con todo el cuerpo visible, para que el sistema pueda identificar todas las articulaciones relevantes para el cálculo de los ángulos. Los movimientos deben ser lo más naturales posibles para obtener mediciones precisas.

Se debe revisar periódicamente el funcionamiento del sistema, especialmente después de actualizaciones o cambios en el entorno. Verificar que los drivers de la cámara y los componentes de procesamiento estén actualizados y funcionando correctamente.

La implementación de este tipo de sistemas con visiones artificiales puede proporcionar al médico información detallada y organizada por lo que puede optimizar el registro médico que contiene los avances y datos importantes de la rehabilitación en adultos mayores, se recomienda mejorar el sistema a futuro para que se pueda evaluar todos los tipos de ángulo que se miden dentro de la goniometría.

7. Bibliografía

Andriluka, M., Pishchulin, L., Gehler, P., & Schiele, B. (2014). *2D Human Pose Estimation: New Benchmark and State of the Art Analysis*.

Araujo, G. G. C., Pontes-Silva, A., Leal, P. da C., Gomes, B. S., Reis, M. L., de Mello Pereira Lima, S. K., Fidelis-de-Paula-Gomes, C. A., & Dibai-Filho, A. V. (2024). Goniometry and fleximetry measurements to assess cervical range of motion in individuals with chronic neck pain: a validity and reliability study. *BMC Musculoskeletal Disorders*, *25*(1). https://doi.org/10.1186/s12891-024-07775-6

Arévalo-Espejo, V., Ambrosio, G., & González-Jiménez, J. (2005). OpenCV. La Librería Open Source de Visión Artificial (in Spanish). *Linux Free Magazine*, 141–147.

Basheer, I. A., & Hajmeer, M. (2000). Artificial neural networks: Fundamentals, computing, design, and application. *Journal of Microbiological Methods*, *43*(1), 3–31. https://doi.org/10.1016/S0167-7012(00)00201-3

Cao, Z., Simon, T., Wei, S.-E., & Sheikh, Y. (2017). *Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields* *. https://youtu.be/pW6nZXeWlGM

Castaneda Sanchez, W. A., Polo Escobar, B. R., & Vega Huincho, F. (2023). Artificial neural networks: a measurement of forecast learnings as potential demand. *Universidad Ciencia y Tecnología*, *27*(118), 51–60. https://doi.org/10.47460/uct.v27i118.686

Chollet, F. (2017). *Xception: Deep Learning with Depthwise Separable Convolutions*.

COGNEX. (2016). *INTRODUCCIÓN A LA VISIÓN ARTIFICIAL Una guía para la automatización de procesos y mejorar la calidad*.

Dubey, S., & Dixit, M. (2023). A comprehensive survey on human pose estimation approaches. *Multimedia Systems*, *29*(1), 167–195. https://doi.org/10.1007/S00530-022-00980-0

Gandbhir, V. N., & Cunha, B. (2022). Goniometer. *Elemente der physikalischen und chemischen Krystallographie*, 342–346. https://doi.org/10.1515/9783486746174-022

IBM. (2012). *What is Artificial Intelligence (AI)*. https://www.ibm.com/topics/artificial-intelligence

Intel. (2010). *Procesador Intel® Core™ i5-8300H*. https://www.intel.la/content/www/xl/es/products/sku/134876/intel-core-i58300h-processor-8m-cache-up-to-4-00-ghz/specifications.html

ISO/IEC/IEEE. (2011). *ISO/IEC/IEEE 29148:2011(E), Systems and software engineering — Life cycle processes — Requirements engineering*. www.iso.org

Juan Sánchez Hernández, J. (2023). *Creación de bases de datos en SQLite*.

Kim, T. S., & Reiter, A. (2017). *Interpretable 3D Human Action Analysis with Temporal Convolutional Networks*.

Lecun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. En *Nature* (Vol. 521, Número 7553, pp. 436–444). Nature Publishing Group. https://doi.org/10.1038/nature14539

Li, G., Das, A., Davidson, S., & Furber, S. B. (2021). Article 651141 Citation: Davidson S and Furber SB (2021) Comparison of Artificial and Spiking Neural Networks on Digital Hardware. *Front. Neurosci*, *15*, 651141. https://doi.org/10.3389/fnins.2021.651141

Lubinus Badillo, F., Rueda Hernández, C. A., Marconi Narváez, B., & Arias Trillos, Y. E. (2021). Redes neuronales convolucionales: un modelo de Deep Learning en imágenes diagnósticas. Revisión de tema. *Revista colombiana de radiología*, *32*(3), 5591–5599. https://doi.org/10.53903/01212095.161

MATLAB. (2015). *What Is Deep Learning? | How It Works, Techniques & Applications - MATLAB & Simulink*. https://www.mathworks.com/discovery/deep-learning.html

Mccarthy, J. (2007). *WHAT IS ARTIFICIAL INTELLIGENCE?* http://www-formal.stanford.edu/jmc/

McKinnon, C. D., Ehmke, S., Kociolek, A. M., Callaghan, J. P., & Keir, P. J. (2021). Wrist Posture Estimation Differences and Reliability Between Video Analysis and Electrogoniometer Methods. *Human Factors*, *63*(7). https://doi.org/10.1177/0018720820923839

Milanese, S., Gordon, S., Buettner, P., Flavell, C., Ruston, S., Coe, D., O'Sullivan, W., & McCormack, S. (2014). Reliability and concurrent validity of knee angle measurement: Smart phone app versus universal goniometer used by experienced and novice clinicians. *Manual Therapy*, *19*(6), 569–574. https://doi.org/10.1016/J.MATH.2014.05.009

Morán, E. (2018). *Diseño e implementación de una red neuronal convolucional para reconocimiento de productos de una empresa de retail*.

NVIDIA. (2008). *Tarjeta gráfica GeForce GTX 1050 | NVIDIA GeForce*. https://www.nvidia.com/es-la/geforce/products/10series/geforce-gtx-1050/

Odemakinde, E. (2023). *Pose Estimation: The Ultimate Overview in 2023 - viso.ai*. https://viso.ai/deep-learning/pose-estimation-ultimate-overview/

Olabe, X. B. (2008). *REDES NEURONALES ARTIFICIALES Y SUS APLICACIONES*.

OPENCV. (2018). *About - OpenCV*. https://opencv.org/about/

OpenPose. (2019). *Biblioteca Openpose: compilación e instalación*. https://github.com/tramper2/openpose/blob/master/doc/installation.md

Pasqual, A. (2003). *Manual De Goniometria*.

Petazzoni, M., & Jaeger, G. (2008). *Atlas of Clinical Goniometry and Radiographic Measurements of the Canine Pelvic Limb* (2ª ed.).

Pineda, J. M. (2022). Modelos predictivos en salud basados en aprendizaje de maquina (machine learning). *Revista Médica Clínica Las Condes*, *33*(6), 583–590. https://doi.org/10.1016/J.RMCLC.2022.11.002

PYTHON. (2015). *What is Python? Executive Summary*. https://www.python.org/doc/essays/blurb/

Rawat Ajay. (2020). *A Review on Python Programming*.

Reusing, M., Brocardo, M., Weber, S., & Villanova, J. (2020). *Goniometric Evaluation and Passive Range of Joint Motion in Chondrodystrophic and Non-Chondrodystrophic Dogs of Different Sizes*. https://doi.org/10.1055/s-0040-1713825

Russell, S., & Norvig, P. (2022). *Artificial Intelligence A Modern Approach* (Fourth). Pearson Education, Inc.

Sarker, I. H. (2021). Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions. *SN Computer Science*, *2*(6). https://doi.org/10.1007/S42979-021-00815-1

Smola, A., & Vishwanathan, S. (2008). *Introduction to Machine Learning*. Cambridge Univiversity Press.

Song, L., Yu, G., Yuan, J., & Liu, Z. (2021). Human pose estimation and its application to action recognition: A survey. *Journal of Visual Communication and Image Representation*, *76*. https://doi.org/10.1016/J.JVCIR.2021.103055

Srinath, K. R. (2017). Python-The Fastest Growing Programming Language. *International Research Journal of Engineering and Technology*. www.irjet.net

Sukari, A. A. A., Singh, S., Bohari, M. H., Idris, Z., Ghani, A. R. I., & Abdullah, J. M. (2021). Examining the range of motion of the cervical spine: Utilising different bedside instruments. *Malaysian Journal of Medical Sciences*, *28*(2), 100–105. https://doi.org/10.21315/mjms2021.28.2.9

Taboadela, C. H. (2007). *GONIOMETRÍA Una herramienta para la evaluación de las incapacidades laborales*. www.asociart.com.ar

Tozzo, M. C., Ansanello, W., Martins, J., Zatiti, S. C. A., & de Oliveira, A. S. (2021). Inclinometer Reliability for Shoulder Ranges of Motion in Individuals With Subacromial Impingement Syndrome. *Journal of Manipulative and Physiological Therapeutics*, *44*(3), 236–243. https://doi.org/10.1016/J.JMPT.2020.12.001

Youdas, J., Garret, T., Suman, V., & Borag, C. (2011). *Nomal Range of Motion of the Cervical Spine: An Initial Goniometric Study*.

8. Anexos

ANEXO A

Requisitos y librerías que deben ser instaladas

```
argparse
dill
fire
matplotlib
numba
psutil
pycocotools
requests
scikit-image
scipy
slidingwindow
tqdm
tensorpack
jinja2
pdfkit
greenlet==3.1.1
importlib-metadata==6.7.0
numpy==1.21.6
opencv-python==4.10.0.84
Pillow==9.5.0
SQLAlchemy==2.0.36
SQLAlchemy-Utils==0.41.2
ttkbootstrap==1.10.1
typing_extensions==4.6.1
wincertstore==0.2
zipp==3.15.0
```

Código Inicio del programa

Archivo main.py (Inicia el programa):

```
from gui.views.patient import PatientView
import ttkbootstrap as ttk
import tkinter as tk
from gui.models.engine import DatabaseEngine
from gui.models.models import Patient


db = DatabaseEngine()
root = ttk.Window(themename="darkly")
application=PatientView(db, root)


root.mainloop() # Start the GUI
```

Archivo engine from gui.models.engine (Base de datos)

```
from sqlalchemy import create_engine
from sqlalchemy_utils import database_exists, create_database
from gui.models.models import Base
from sqlalchemy.orm import Session
from sqlalchemy import select, delete, update

class DatabaseEngine:
    __engine = None
    __session = None
    def __init__(self) -> None:
        self.__engine = create_engine("sqlite:///db.sqlite", echo=True)
        self.__session = Session(self.__engine)

    def createAll(self):
        engine = self.__engine
        Base.metadata.create_all(engine)
        if not database_exists(engine.url):
            create_database(engine.url)

    def getSession(self):
        return self.__session

    def save(self, elements):
        session = self.getSession()
        session.add_all(elements)
        session.commit()

    def simpleQuery(self, model, filter):
        session = self.getSession()
        stmt = select(model)
        if(filter is not None):
            stmt = stmt.where(filter)
        return session.scalars(stmt)

    def update(self, model, filter, args):
        session = self.getSession()
        stmt = update(model).values(args)

        if(filter is not None):
            stmt = stmt.where(filter)



       session.execute(stmt)
       session.commit()

   def delete(self, model, filter):
       session = self.getSession()
       stmt = delete(model)
       if(filter is None):
           return None
       stmt = stmt.where(filter)
       session.execute(stmt)
       session.commit()
```

Archivo models from gui.models.models (Base de datos)

```
from typing import List
from typing import Optional
from sqlalchemy import ForeignKey
from sqlalchemy import UniqueConstraint
from sqlalchemy import String
from sqlalchemy.orm import DeclarativeBase
from sqlalchemy.orm import Mapped
from sqlalchemy.orm import mapped_column
from sqlalchemy.orm import relationship


class Base(DeclarativeBase):
    pass

class Patient(Base):
    __tablename__ = "patient"
    id: Mapped[int] = mapped_column(primary_key=True)
    ci: Mapped[str] = mapped_column(String(13), unique=True)
    firstname: Mapped[str] = mapped_column(String(50))
    lastname: Mapped[str] = mapped_column(String(50))
    genre: Mapped[str] = mapped_column(String(1))
    profession: Mapped[str] = mapped_column(String(30))
    birthDate: Mapped[str] = mapped_column(String(10))
    measurements: Mapped[List["Measurement"]] = relationship(
        back_populates="patient", cascade="all, delete-orphan"
    )
    def __repr__(self) -> str:
        return f"Patient(id={self.id!r}, firstname={self.firstname!r}, lastname={self.lastname!r})"


class Measurement(Base):
    __tablename__ = "measurement"
    id: Mapped[int] = mapped_column(primary_key=True)
    flexion: Mapped[float] = mapped_column()
    extension: Mapped[float] = mapped_column()
    right_tilt: Mapped[float] = mapped_column()
    left_tilt: Mapped[float] = mapped_column()
    right_roration: Mapped[float] = mapped_column()
    left_roration: Mapped[float] = mapped_column()
    flexion_image: Mapped[str] = mapped_column(String(50), nullable=True)
    extension_image: Mapped[str] = mapped_column(String(50), nullable=True)
    right_tilt_image: Mapped[str] = mapped_column(String(50), nullable=True)
    left_tilt_image: Mapped[str] = mapped_column(String(50), nullable=True)
    right_roration_image: Mapped[str] = mapped_column(String(50), nullable=True)
    left_roration_image: Mapped[str] = mapped_column(String(50), nullable=True)
    patient: Mapped["Patient"] = relationship(back_populates="measurements")
    patient_id: Mapped[int] = mapped_column(ForeignKey("patient.id"))
    def __repr__(self) -> str:
        return f"Measurement(id={self.id!r})"
```

El archivo patient inicia la interfaz gráfica, se detalla en el Anexo C.

ANEXO B

Código Cálculo de ángulos

```
from image_capture import takePicture
import cv2
import numpy as np
import math
import os
import uuid

pointsDictionary = {
    "nose": 0,
    "neck": 1,
    "rightShoulder": 2,
    "leftShoulder": 5,
    "rightEar": 16,
    "leftEar": 17
}

def getPoints(pointNames):
    points = []
    for name in pointNames:
        points.append(pointsDictionary[name])
    return points

def getCordinatesOfPoints(points, centers):
    pointCenters = []
    try:
        for point in points:
            if(centers[point] != ''):
                pointCenters.append(centers[point])
            else:
                return None
    except:
        return None

    return pointCenters

def validateOption(option):
    if(option != '1' and option != '2' ):
        print('! Adiós !')
        exit()



def putTextOnImage(image, text = '',  color = (0, 0, 0), scale = 0.5, x = 0, y = 0, thickness = 2):
    cv2.putText( image,text,(x, y), cv2.FONT_HERSHEY_SIMPLEX, scale, color, thickness )

def calcaluateAngle(centralPoint, point1, point2):
    A = point1
    B = centralPoint
    C = point2

    # VECTORES

    BA = np.array([A[0] - B[0], A[1] - B[1]])
    BC = np.array([C[0] - B[0], C[1] - B[1]])

    # MODULOS O DISTANCIA ENTRE PUNTOS

    MOD_BA = math.sqrt( (A[0] - B[0])**2 + (A[1] - B[1])**2)
    MOD_BC = math.sqrt( (C[0] - B[0])**2 + (C[1] - B[1])**2)

    # PRODUCTO ESCALAR ENTRE VECTORES V1@V2
    # FORMULA -> V1.V2 = |V1|.|V2|*cos(theta)

    P_ESCALAR_BA_BC = BA@BC
```

```
    # CALCULO DEL ANGULO THETA 1 -> LADO DERECHO
    # CALCULO DEL ANGULO THETA 2 -> LADO IZQUIERDO

    COS_THETHA1 = (P_ESCALAR_BA_BC)/(MOD_BA*MOD_BC)

    theta1 = math.degrees(math.acos(COS_THETHA1))
    return theta1

def calculateDistanceBetween2Points(pointA, pointB):
    A = pointA
    B = pointB
    distance = math.sqrt( (B[0] - A[0])**2 + (B[1] - A[1])**2)
    return distance


def getAngle(movement_type: str):

    image, centers = takePicture(model = 'mobilenet_thin', resize = '432x368')
    path = './pictures/'
    if(not os.path.isdir(path)):
        os.makedirs(path)

    imageName = path + str(uuid.uuid4()) + ".png"

    cv2.imwrite(imageName, img=image)

    thetha = None

    if(movement_type == 'flexion' or movement_type == 'extension'):
        points = getPoints(['rightShoulder', 'nose'])

        cordinatesOfPoints = getCordinatesOfPoints(points, centers)


        if(cordinatesOfPoints != None):
            cordinatesOfPoints.append(
                (cordinatesOfPoints[0][0], cordinatesOfPoints[0][1] - 1)
            )

            print('Points', points)
            print('cordinatesOfPoints', cordinatesOfPoints)

            thetha = calcaluateAngle(cordinatesOfPoints[0], cordinatesOfPoints[1], cordinatesOfPoints[2])

    elif(movement_type == 'right_tilt'):

        points = getPoints(['neck', 'nose', 'rightShoulder'])

        cordinatesOfPoints = getCordinatesOfPoints(points, centers)

        if(cordinatesOfPoints != None):
            print('Points', points)
            print('cordinatesOfPoints', cordinatesOfPoints)

            thetha = 90 - calcaluateAngle(cordinatesOfPoints[0], cordinatesOfPoints[1], cordinatesOfPoints[2])


    elif(movement_type == 'left_tilt'):

        points = getPoints(['neck', 'nose', 'leftShoulder'])

        cordinatesOfPoints = getCordinatesOfPoints(points, centers)

        if(cordinatesOfPoints != None):
            print('Points', points)
            print('cordinatesOfPoints', cordinatesOfPoints)

            thetha = 90 - calcaluateAngle(cordinatesOfPoints[0], cordinatesOfPoints[1], cordinatesOfPoints[2])



    elif(movement_type == 'right_rotation'):
        points = getPoints(['rightShoulder', 'nose'])

        print("\n Caputura 1: Posición recta mirando hacia el frente.")

        cordinatesOfPoints1 = getCordinatesOfPoints(points, centers)

        print("\n Caputura 2: Posición recta mirando hacia la derecha")

        image, centers = takePicture(model = 'mobilenet_thin', resize = '432x368')

        cv2.imwrite(imageName, img=image)

        cordinatesOfPoints2 = getCordinatesOfPoints(points, centers)
        if(cordinatesOfPoints1 != None and cordinatesOfPoints2 != None):
            hyp = abs(cordinatesOfPoints1[0][0] - cordinatesOfPoints1[1][0])
            ady = abs(cordinatesOfPoints2[0][0] - cordinatesOfPoints2[1][0])
```

```
            thetha = 90 - math.degrees(math.acos( ady/hyp ) )

    elif(movement_type == 'left_rotation'):
        points = getPoints(['leftShoulder', 'nose'])

        print("\n Caputura 1: Posición recta mirando hacia el frente.")

        cordinatesOfPoints1 = getCordinatesOfPoints(points, centers)

        print("\n Caputura 2: Posición recta mirando hacia la izquierda")

        image, centers = takePicture(model = 'mobilenet_thin', resize = '432x368')

        cv2.imwrite(imageName, img=image)

        cordinatesOfPoints2 = getCordinatesOfPoints(points, centers)
        if(cordinatesOfPoints1 != None and cordinatesOfPoints2 != None):
            hyp = abs(cordinatesOfPoints1[0][0] - cordinatesOfPoints1[1][0])
            ady = abs(cordinatesOfPoints2[0][0] - cordinatesOfPoints2[1][0])

            thetha = 90 - math.degrees(math.acos( ady/hyp ) )

    print('Angle: ', thetha)
    cv2.destroyAllWindows()
    return thetha, imageName
```

ANEXO C

Diseño de la Interfaz Gráfica

GUI “patient”

```
CRUD-PRODUCTOS
-Registro de Pacientes
-Guardar en bd SQlite

"""
import ttkbootstrap as ttk
from ttkbootstrap import Label, Frame, Labelframe, Button, Entry, DateEntry
from ttkbootstrap.constants import *
import tkinter as tk
import tkinter.messagebox as messagebox

#Python image Library
from PIL import ImageTk, Image
import sqlite3

from gui.models.models import Patient
from gui.models.engine import DatabaseEngine
from gui.views.measurement import MeasurementView

class PatientView:
    db_name='database_proyecto.db'
    current_patient: Patient = None

    def __init__(self, engine: DatabaseEngine, view: ttk.Window):
        self.engine = engine
        self.window=view
        self.window.title("APLICACION")
        self.window.geometry("800x670")
        self.window.resizable(0,0)
        self.window.config(bd=10)

        "--------------- Titulo --------------------"
        Label(view, text="REGISTRO DE PACIENTES", font=("Comic Sans", 17,"bold")).pack()



        "--------------- Frame marco --------------------"
        formFrame = Labelframe(view, text="Informacion del paciente")
        formFrame.config(border=2)
        formFrame.pack()
```

```
    "--------------- Formulario --------------------"
    Label(formFrame,text="Nombres: ").grid(row=0,column=0,sticky='s',padx=5,pady=8)
    self.firstname=Entry(formFrame,width=25)
    self.firstname.focus()
    self.firstname.grid(row=0, column=1, padx=5, pady=8)

    Label(formFrame,text="Apellidos: ").grid(row=1,column=0,sticky='s',padx=5,pady=8)
    self.lastname=Entry(formFrame,width=25)
    self.lastname.grid(row=1, column=1, padx=5, pady=8)

    Label(formFrame,text="Género: ").grid(row=2,column=0,sticky='s',padx=5,pady=9)
    self.genre=ttk.Combobox(formFrame,values=["Masculino","Femenino"], width=22,state="readonly")
    self.genre.current(0)
    self.genre.grid(row=2,column=1,padx=5,pady=0)


    Label(formFrame,text="Cédula: ").grid(row=0,column=2,sticky='s',padx=5,pady=8)
    self.ci=Entry(formFrame,width=15)
    self.ci.grid(row=0, column=3, padx=5, pady=8)

    Label(formFrame,text="Fecha Nacimiento: ").grid(row=1,column=2,sticky='s',padx=5,pady=8)
    self.birthDate=DateEntry(formFrame,width=15)
    self.birthDate.grid(row=1, column=3, pady=8)

    Label(formFrame,text="Profesión: ").grid(row=2,column=2,sticky='s',padx=10,pady=8)
    self.profession=Entry(formFrame,width=15)
    self.profession.grid(row=2, column=3, pady=8)

    "--------------- Frame botones --------------------"
    buttonFrame=Frame(view)
    buttonFrame.pack()

    "--------------- Botones --------------------"
    boton_registrar=Button(buttonFrame,text="GUARDAR",command=self.savePatient,width=10).grid(row=0, column=1,
    boton_editar=Button(buttonFrame,text="EDITAR",command=self.editPatient ,width=10).grid(row=0, column=2, pa
    boton_diagnosticar=Button(buttonFrame,text="DIAGNOSTICAR",command=self.makeDiagnosis ,width=10, style=SUCC
    boton_eliminar=Button(buttonFrame,text="ELIMINAR",command=self.deletePatient ,width=10, style=DANGER).grid



    "--------------- Tabla --------------------"
    self.tree=ttk.Treeview(height=13, columns=("columna1","columna2","columna3","columna4","columna5"))
    self.tree.heading("#0",text='Cédula', anchor=CENTER)
    self.tree.column("#0", width=100, minwidth=75, stretch=NO)

    self.tree.heading("columna1",text='Nombre', anchor=CENTER)
    self.tree.column("columna1", width=150, minwidth=75, stretch=NO)

    self.tree.heading("columna2",text='Apellido', anchor=CENTER)
    self.tree.column("columna2", width=150, minwidth=75, stretch=NO)

    self.tree.heading("columna3",text='Género', anchor=CENTER)
    self.tree.column("columna3", width=80, minwidth=60, stretch=NO)

    self.tree.heading("columna4",text='Fecha Nac.', anchor=CENTER)
    self.tree.column("columna4", width=90, minwidth=60, stretch=NO)

    self.tree.heading("columna5",text='Profesión', anchor=CENTER)
    self.tree.column("columna5", width=100, minwidth=60, stretch=NO)

    self.tree.pack()

    self.getPatients()

def getPatients(self):

    records = self.tree.get_children()
    for element in records:
        self.tree.delete(element)

    results = self.engine.simpleQuery(Patient, None)
    for p in results:
        self.tree.insert("",0, text=p.ci ,values=(p.firstname,p.lastname,p.genre, p.birthDate, p.profession))
```

```
def savePatient(self):
    if self.validateForm() and self.current_patient == None:
        new_patient = Patient(
            firstname = self.firstname.get(),
            lastname = self.lastname.get(),
            ci = self.ci.get(),
            genre = self.genre.get(),
            profession = self.profession.get(),
            birthDate = self.birthDate.entry.get()
        )

        self.engine.save([new_patient])
    elif self.validateForm() and self.current_patient != None:

        self.engine.update(Patient, Patient.ci.__eq__(self.current_patient.ci), {
                            "firstname" : self.firstname.get(),
                            "lastname" : self.lastname.get(),
                            "ci" : self.ci.get(),
                            "genre" : self.genre.get(),
                            "birthDate" : self.birthDate.entry.get(),
                            "profession" : self.profession.get()
                            }
                        )
    self.getPatients()
    self.resetForm()


def editPatient(self):
    try:
        self.tree.item(self.tree.selection())['text'][0]
    except IndexError as e:
        messagebox.showerror("ERROR","Porfavor selecciona un elemento")
        return
    db = self.engine
    cedula=self.tree.item(self.tree.selection())['text']
    self.current_patient = db.simpleQuery(Patient, Patient.ci.__eq__(cedula)).one()
    self.fillForm()


def makeDiagnosis(self):
    try:
        self.tree.item(self.tree.selection())['text'][0]
    except IndexError as e:
        messagebox.showerror("ERROR","Porfavor selecciona un elemento")
        return
    db = self.engine
    cedula=self.tree.item(self.tree.selection())['text']
    patient = db.simpleQuery(Patient, Patient.ci.__eq__(cedula)).one()

    root = ttk.Toplevel()
    MeasurementView(self.engine, root, patient)
    #self.window.destroy()
    root.mainloop()

def fillForm(self):
    self.firstname.delete(0, END)
    self.lastname.delete(0, END)
    self.ci.delete(0, END)
    self.genre.delete(0, END)
    self.profession.delete(0, END)
    self.birthDate.entry.delete(0, END)
    print('Patient: ',self.current_patient)
    self.firstname.insert(0, self.current_patient.firstname)
    self.lastname.insert(0, self.current_patient.lastname)
    self.ci.insert(0, self.current_patient.ci)
    self.genre.insert(0, self.current_patient.genre)
    self.profession.insert(0, self.current_patient.profession)
    self.birthDate.entry.insert(0, self.current_patient.birthDate)

def validateForm(self):
    if len(self.firstname.get()) !=0 and len(self.lastname.get()) !=0 and len(self.genre.get())
        return True
    else:
         messagebox.showerror("ERROR", "Complete todos los campos del formulario")
```

```
    def resetForm(self):
        self.current_patient = None
        self.firstname.delete(0, END)
        self.lastname.delete(0, END)
        self.ci.delete(0, END)
        self.genre.delete(0, END)
        self.profession.delete(0, END)
        self.birthDate.entry.delete(0, END)

    def deletePatient(self):
        try:
            self.tree.item(self.tree.selection())['text'][0]
        except IndexError as e:
            messagebox.showerror("ERROR","Porfavor selecciona un elemento")
            return
        ci = self.tree.item(self.tree.selection())['text']
        db = self.engine
        db.delete(Patient, Patient.ci.__eq__(ci))
        self.getPatients()

#if __name__ == '__main__':
    #ventana_producto=Tk()
    #application=Producto(ventana_producto)
    #ventana_producto.mainloop()
```

Resultado:

GUI Measurement

```
import ttkbootstrap as ttk
from ttkbootstrap import Label, Frame, Labelframe, Button, Entry
from ttkbootstrap.constants import *
import tkinter as tk
import tkinter.messagebox as messagebox


#Python image Library
from PIL import ImageTk, Image
import sqlite3

#Models
from gui.models.models import Patient, Measurement
from gui.models.engine import DatabaseEngine

from gui.reports.report_maker import createReport



#TakePicture MODULES
import cv2
from PIL import Image, ImageTk

from gui.angle_calc import getAngle

class MeasurementView:
    db_name='database_proyecto.db'
    current_patient: Patient = None
    current_measurement: Measurement = Measurement()

    def __init__(self, engine: DatabaseEngine, view: ttk.Window, patient: Patient):
        self.current_patient = patient
        self.engine = engine
        self.window=view
        self.window.title("APLICACION")
        #self.window.geometry("800x670")
        #self.window.resizable(0,0)
        self.window.config(bd=10)
        number_validator = self.window.register(validate_angle)

"---------------- Frames ---------------------"

self.titleFrame = Frame(self.window)
self.titleFrame.config(border=2)
self.titleFrame.grid(row=0, column=0, columnspan=2, padx=20, pady=20)

self.patientFrame = Labelframe(self.window, text="Datos del paciente")
self.patientFrame.config(border=2)
self.patientFrame.grid(row=1, column=0, padx=20, pady=20)

self.tableFrame=Frame(view)
self.tableFrame.grid(row=2, column=0, columnspan=1)
tableFrame = self.tableFrame

self.formFrame = Labelframe(self.window, text="Datos del diagnóstico")
self.formFrame.config(border=2)
self.formFrame.grid(row=3, column=0, padx=20, pady=20)

self.buttonFrame=Frame(view)
self.buttonFrame.grid(row=4, column=0, columnspan=2)
buttonFrame = self.buttonFrame

self.imageFrame = Labelframe(self.window, text="Imagenes")
self.imageFrame.config(border=2)
self.imageFrame.grid(row=1, column=1, padx=20, pady=20, rowspan=3)

formFrame = self.formFrame



"--------------- Titulo --------------------"

Label(self.titleFrame, text="Diagnóstico de movimientos cervicales", font=("Comic Sans", 14,"bold")).pack()

"--------------- Paciente --------------------"

Label(self.patientFrame, text="Nombre: {} {}".format(patient.firstname, patient.lastname), font=("Comic Sans", 11,"bold")).grid(row=0, column=0, padx=20)
Label(self.patientFrame, text="Cédula: {}".format(patient.ci), font=("Comic Sans", 11,"bold")).grid(row=0, column=1, padx=20)
```

```
"--------------- Imagen Frames--------------------"


self.flexionFrame = Labelframe(self.imageFrame, text="Flexión")
self.flexionFrame.config(border=2)
self.flexionFrame.grid(row=0, column=0, padx=20, pady=20)

self.extensionFrame = Labelframe(self.imageFrame, text="Extensión")
self.extensionFrame.config(border=2)
self.extensionFrame.grid(row=1, column=0, padx=20, pady=20)

self.rightTiltFrame = Labelframe(self.imageFrame, text="Inc. Derecha")
self.rightTiltFrame.config(border=2)
self.rightTiltFrame.grid(row=0, column=1, padx=20, pady=20)

self.leftTiltFrame = Labelframe(self.imageFrame, text="Inc. Izquierda")
self.leftTiltFrame.config(border=2)
self.leftTiltFrame.grid(row=1, column=1, padx=20, pady=20)

self.rightRotationFrame = Labelframe(self.imageFrame, text="Rot. Derecha")
self.rightRotationFrame.config(border=2)
self.rightRotationFrame.grid(row=0, column=2, padx=20, pady=20)

self.leftRotationFrame = Labelframe(self.imageFrame, text="Rot. Izquierda")
self.leftRotationFrame.config(border=2)
self.leftRotationFrame.grid(row=1, column=2, padx=20, pady=20)


self.flexionImage = None
self.flexionLabel = tk.Label(self.flexionFrame)
self.flexionLabel.image = self.flexionImage
self.flexionLabel.configure(image=None)
self.flexionLabel.grid(row=0, column=0)

self.extensionImage = None
self.extensionLabel = tk.Label(self.extensionFrame)
self.extensionLabel.image = self.extensionImage
self.extensionLabel.configure(image=None)
self.extensionLabel.grid(row=0, column=0)


 self.rightTiltImage = None
 self.rightTiltLabel = tk.Label(self.rightTiltFrame)
 self.rightTiltLabel.image = self.rightTiltImage
 self.rightTiltLabel.configure(image=None)
 self.rightTiltLabel.grid(row=0, column=0)

 self.leftTiltImage = None
 self.leftTiltLabel = tk.Label(self.leftTiltFrame)
 self.leftTiltLabel.image = self.leftTiltImage
 self.leftTiltLabel.configure(image=None)
 self.leftTiltLabel.grid(row=0, column=0)

 self.rightRotationImage = None
 self.rightRotationLabel = tk.Label(self.rightRotationFrame)
 self.rightRotationLabel.image = self.rightRotationImage
 self.rightRotationLabel.configure(image=None)
 self.rightRotationLabel.grid(row=0, column=0)

 self.leftRotationImage = None
 self.leftRotationLabel = tk.Label(self.leftRotationFrame)
 self.leftRotationLabel.image = self.leftRotationImage
 self.leftRotationLabel.configure(image=None)
 self.leftRotationLabel.grid(row=0, column=0)
```

```
        "--------------- Botones --------------------"
        takePictureButton = Button(buttonFrame,text="REVISAR",command=self.reviewMeasurement,width=10, style=SUCCESS)
        takePictureButton.grid(row=0, column=1, padx=10, pady=15)

        saveButton = Button(buttonFrame,text="GUARDAR",command=self.saveMeasurement,width=10, style=PRIMARY)
        saveButton.grid(row=0, column=2, padx=10, pady=15)

        saveButton = Button(buttonFrame,text="REPORTE",command=self.makePDFReport,width=10, style=INFO)
        saveButton.grid(row=0, column=3, padx=10, pady=15)

        saveButton = Button(buttonFrame,text="BORRAR",command=self.deleteMeasurement,width=10, style=DANGER)
        saveButton.grid(row=0, column=4, padx=10, pady=15)
        #self.getPatients()

        "---------------- Tabla de diagnosticos -------------------"
        self.tree=ttk.Treeview(tableFrame, height=10, columns=("columna0","columna1","columna2","columna3","columna4'
        self.tree.heading("#0",text='Nro.', anchor=CENTER)
        self.tree.column("#0", width=50, minwidth=50, stretch=NO)

        self.tree.heading("columna0",text='Paciente', anchor=CENTER)
        self.tree.column("columna0", width=150, minwidth=150, stretch=NO)

        self.tree.heading("columna1",text='Flexión', anchor=CENTER)
        self.tree.column("columna1", width=50, minwidth=50, stretch=NO)

        self.tree.heading("columna2",text='Extensión', anchor=CENTER)
        self.tree.column("columna2", width=50, minwidth=50, stretch=NO)

        self.tree.heading("columna3",text='Inc. Der.', anchor=CENTER)
        self.tree.column("columna3", width=50, minwidth=50, stretch=NO)

        self.tree.heading("columna4",text='Inc. Izq.', anchor=CENTER)
        self.tree.column("columna4", width=50, minwidth=50, stretch=NO)

        self.tree.heading("columna5",text='Rot. Der.', anchor=CENTER)
        self.tree.column("columna5", width=50, minwidth=50, stretch=NO)

        self.tree.heading("columna6",text='Rot. Izq.', anchor=CENTER)
        self.tree.column("columna6", width=50, minwidth=50, stretch=NO)



    self.tree.column("columna7", width=0)

    self.tree.pack()
    self.getPatientMeasurements()


def makeMeasurement(self, movement_type: str):
    thetha, imgName = getAngle(movement_type)
    img = Image.open(imgName)
    return self.resizeImage(img), imgName, thetha

def resizeImage(self, img):
    newWidth = 125

    oldWidth = img.width
    oldHeight = img.height

    newHeight = int(oldHeight*newWidth/oldWidth)

    newSize = [newWidth, newHeight]
    return ImageTk.PhotoImage(img.resize(newSize))

def setLabelImage(self, label: Label, imgPointer: ImageTk.PhotoImage, imgValue: ImageTk.PhotoImage):
    imgPointer = imgValue
    label.image = imgPointer
    label.configure(image=imgPointer)

def captureFlexion(self):
    #getFlexionExtensionAngle()
    img, imgName, thetha = self.makeMeasurement('flexion')

    self.flexionImage = img
    self.flexionLabel.image = self.flexionImage
    self.flexionLabel.configure(image=self.flexionImage)
    self.flexion.delete(0,END)
    self.flexion.insert(0, thetha)
    self.current_measurement.flexion_image = imgName
```

```
 def captureExtension(self):
     img, imgName, thetha = self.makeMeasurement('extension')
     self.extensionImage = img
     self.extensionLabel.image = self.extensionImage
     self.extensionLabel.configure(image=self.extensionImage)
     self.extension.delete(0,END)
     self.extension.insert(0,thetha)
     self.current_measurement.extension_image = imgName

 def captureRightTilt(self):
     img, imgName, thetha = self.makeMeasurement('right_tilt')
     self.rightTiltImage = img
     self.rightTiltLabel.image = self.rightTiltImage
     self.rightTiltLabel.configure(image=self.rightTiltImage)
     self.right_tilt.delete(0,END)
     self.right_tilt.insert(0,thetha)
     self.current_measurement.right_tilt_image = imgName

 def captureLeftTilt(self):
     img, imgName, thetha = self.makeMeasurement('left_tilt')
     self.leftTiltImage = img
     self.leftTiltLabel.image = self.leftTiltImage
     self.leftTiltLabel.configure(image=self.leftTiltImage)
     self.left_tilt.delete(0,END)
     self.left_tilt.insert(0,thetha)
     self.current_measurement.left_tilt_image = imgName

 def captureRightRotation(self):
     img, imgName, thetha = self.makeMeasurement('right_rotation')
     self.rightRotationImage = img
     self.rightRotationLabel.image = self.rightRotationImage
     self.rightRotationLabel.configure(image=self.rightRotationImage)
     self.right_roration.delete(0,END)
     self.right_roration.insert(0,thetha)
     self.current_measurement.right_roration_image = imgName


def captureLeftRotation(self):
    img, imgName, thetha = self.makeMeasurement('left_rotation')
    self.leftRotationImage = img
    self.leftRotationLabel.image = self.leftRotationImage
    self.leftRotationLabel.configure(image=self.leftRotationImage)
    self.left_roration.delete(0,END)
    self.left_roration.insert(0,thetha)
    self.current_measurement.left_roration_image = imgName

def saveMeasurement(self):

    self.current_measurement.flexion = self.flexion.get()
    self.current_measurement.extension = self.extension.get()
    self.current_measurement.right_tilt = self.right_tilt.get()
    self.current_measurement.left_tilt = self.left_tilt.get()
    self.current_measurement.right_roration = self.right_roration.get()
    self.current_measurement.left_roration = self.left_roration.get()
    self.current_measurement.patient = self.current_patient

    self.current_patient.measurements.append(self.current_measurement)

    self.engine.save([self.current_measurement])
    self.getPatientMeasurements()
    self.resetForm()

def getPatientMeasurements(self):

    records = self.tree.get_children()
    for element in records:
        self.tree.delete(element)

    results = self.current_patient.measurements
```

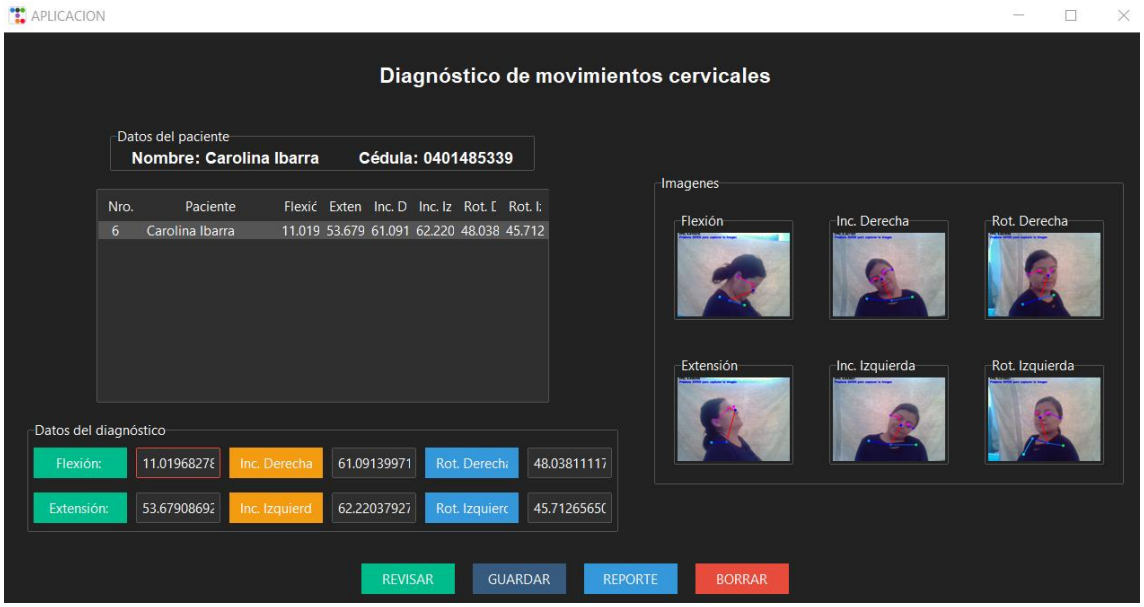
```
    p = self.current_patient
    fullname = "{} {}".format(p.firstname, p.lastname)
    for m in results:
        self.tree.insert("",0, text=str(m.id) ,values=(fullname, m.flexion, m.extension, m.right_tilt, m

def reviewMeasurement(self):

    try:
        self.tree.item(self.tree.selection())['text'][0]
    except IndexError as e:
        messagebox.showerror("ERROR","Porfavor selecciona un elemento")
        return

    self.resetForm()

    id = self.tree.item(self.tree.selection())['text']
    db = self.engine
    measurement: Measurement =  db.simpleQuery(Measurement, Measurement.id.__eq__(id)).one()
    self.current_measurement = measurement
    self.flexion.delete(0,END)
    self.flexion.insert(0,str(measurement.flexion))

    self.extension.delete(0,END)
    self.extension.insert(0,str(measurement.extension))

    self.right_tilt.delete(0,END)
    self.right_tilt.insert(0,str(measurement.right_tilt))

    self.left_tilt.delete(0,END)
    self.left_tilt.insert(0,str(measurement.left_tilt))

    self.right_roration.delete(0,END)
    self.right_roration.insert(0,str(measurement.right_roration))

    self.left_roration.delete(0,END)
    self.left_roration.insert(0,str(measurement.left_roration))



    if(measurement.flexion_image):
        flexionImage = self.resizeImage(Image.open(measurement.flexion_image))
        self.setLabelImage(self.flexionLabel, self.flexionImage, flexionImage)
    if(measurement.extension_image):
        extension_image = self.resizeImage(Image.open(measurement.extension_image))
        self.setLabelImage(self.extensionLabel, self.extensionImage, extension_image)
    if(measurement.right_tilt_image):
        right_tilt_image = self.resizeImage(Image.open(measurement.right_tilt_image))
        self.setLabelImage(self.rightTiltLabel, self.rightTiltImage, right_tilt_image)
    if(measurement.left_tilt_image):
        left_tilt_image = self.resizeImage(Image.open(measurement.left_tilt_image))
        self.setLabelImage(self.leftTiltLabel, self.leftTiltImage, left_tilt_image)
    if(measurement.right_roration_image):
        right_roration_image = self.resizeImage(Image.open(measurement.right_roration_image))
        self.setLabelImage(self.rightRotationLabel, self.rightRotationImage, right_roration_image)
    if(measurement.left_roration_image):
        left_roration_image = self.resizeImage(Image.open(measurement.left_roration_image))
        self.setLabelImage(self.leftRotationLabel, self.leftRotationImage, left_roration_image)

def validateForm(self):
    if len(self.firstname.get()) !=0 and len(self.lastname.get()) !=0 and len(self.genre.get()) !=0 and l
        return True
    else:
         messagebox.showerror("ERROR", "Complete todos los campos del formulario")

def resetForm(self):
    self.current_measurement = Measurement()

    self.flexion.delete(0,END)
    self.flexion.insert(0,'')

    self.extension.delete(0,END)
    self.extension.insert(0,'')

    self.right_tilt.delete(0,END)
    self.right_tilt.insert(0,'')
```

```
   def resetImages(self):

       self.flexionImage = None
       self.flexionLabel.image = self.flexionImage
       self.flexionLabel.configure(image=self.flexionImage)

       self.extensionImage = None
       self.extensionLabel.image = self.extensionImage
       self.extensionLabel.configure(image=self.extensionImage)

       self.rightTiltImage = None
       self.rightTiltLabel.image = self.rightTiltImage
       self.rightTiltLabel.configure(image=self.rightTiltImage)

       self.leftTiltImage = None
       self.leftTiltLabel.image = self.leftTiltImage
       self.leftTiltLabel.configure(image=self.leftTiltImage)

       self.rightRotationImage = None
       self.rightRotationLabel.image = self.rightRotationImage
       self.rightRotationLabel.configure(image=self.rightRotationImage)

       self.leftRotationImage = None
       self.leftRotationLabel.image = self.leftRotationImage
       self.leftRotationLabel.configure(image=self.leftRotationImage)

   def deleteMeasurement(self):
       try:
           self.tree.item(self.tree.selection())['text'][0]
       except IndexError as e:
           messagebox.showerror("ERROR","Porfavor selecciona un elemento")
           return
       id = self.tree.item(self.tree.selection())['text']
       db = self.engine
       db.delete(Measurement, Measurement.id.__eq__(id))
       self.getPatientMeasurements()
```

```
    def makePDFReport(self):

        if (self.current_measurement.id == None):
            messagebox.showerror("ERROR", "Por favor, primero selecciona un elemento y haz click en REVISAR
            return

        result = createReport(self.current_patient, self.current_measurement)
        if result:
            messagebox.showinfo("INFO","Reporte creado correctamente.")
        else:
            messagebox.showerror("ERROR","Porfavor selecciona un elemento")
        return None
```

```
import uuid


def takePicture():
    cam = cv2.VideoCapture(0)

    while True:

        ret_val, img = cam.read()
        #Rearrang the color channel
        b,g,r = cv2.split(img)
        img = cv2.merge((r,g,b))
        cv2.imshow('TakePicture', img)
        keycode = cv2.waitKey(1)
        if keycode == 27:
            imageName = str(uuid.uuid4()) + ".png"
            cv2.imwrite(imageName, img)
            cv2.destroyAllWindows()
            break

    # Convert the Image object into a TkPhoto object
    return imageName

def validate_angle(x) -> bool:
    """Validates that the input is a angle"""
    isValid = False
    try:
        if float(x) <= 180:
            isValid = True
   except:
       pass

   return isValid
```

Resultado:


APLICACION
Diagnóstico de movimientos cervicales
Datos del paciente
Nombre: Carolina Ibarra
Cédula: 0401485339
Nro.
Paciente
Flexić
Exten
Inc. D
Inc. Iz
Rot. [
Rot. I:
6
Carolina Ibarra
11.019
53.679
61.091
62.220
48.038
45.712
Imagenes
Flexión
Inc. Derecha
Rot. Derecha
Extensión
Inc. Izquierda
Rot. Izquierda
Datos del diagnóstico
Flexión:
11.0196827E
Inc. Derecha
61.09139971
Rot. Derechi
48.03811117
Extensión:
53.6790869:
Inc. Izquierd
62.22037927
Rot. Izquierc
45.7126565(
REVISAR
GUARDAR
REPORTE
BORRAR