UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CARRERA DE INGENIERÍA EN MECATRÓNICA

TRABAJO DE INTEGRACIÓN CURRICULAR PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN MECATRÓNICA

TEMA:

"PROGRAMA PARA EL DISEÑO DE MECANISMOS DE 4 BARRAS BASADO EN UN ALGORITMO EVOLUTIVO"

UTN IBARRA - ECUADOR | FICA

Trabajo de grado previo a la obtención del título de Ingeniero en Mecatrónica

AUTOR: Steven Josué Lanchi Iñiguez

DIRECTOR: MSc. Fernando Vinicio Valencia Aguirre

ASESOR: PhD. Marco Antonio Ciaccia Sortino

Ibarra-Ecuador

2025

UNIVERSIDAD TÉCNICA DEL NORTE
BIBLIOTECA UNIVERSITARIA

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

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:	0104831003		
APELLIDOS Y NOMBRES:	Lanchi Iñiguez Steven Josué		
DIRECCIÓN:	Tulcán, Carchi		
EMAIL:	sjlanchii@utn.edu.ec		
TELÉFONO FIJO:	---------	**TELÉFONO MÓVIL:**	0968712574

DATOS DE LA OBRA	
TÍTULO:	"PROGRAMA PARA EL DISEÑO DE MECANISMOS DE 4 BARRAS BASADO EN UN ALGORITMO EVOLUTIVO"
AUTOR (ES):	Lanchi Iñiguez Steven Josué
FECHA DE APROBACIÓN: DD/MM/AAAA	26/02/2025
PROGRAMA:	■ **PREGRADO** □ **POSGRADO**
TITULO POR EL QUE OPTA:	Ingeniero en Mecatrónica
ASESOR /DIRECTOR:	MSc. Fernando Vinicio Valencia Aguirre

1. CONSTANCIAS

El autor (es) manifiesta (n) 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 (son) el (los) titular (es) de los derechos patrimoniales, por lo que asume (n) la responsabilidad sobre el contenido de la misma y saldrá (n) en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 27 días del mes de febrero de 2025

EL AUTOR:

(Firma).....
Nombre: Lanchi Iñiguez Steven Josué

CERTIFICACIÓN DIRECTOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR

Ibarra, 27 de febrero de 2025

MSc. Fernando Vinicio Valencia Aguirre

DIRECTOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR

CERTIFICA:

Haber revisado el presente informe final del trabajo de titulación, el mismo que se ajusta a las normas vigentes de la Unidad Académica de la Universidad Técnica del Norte; en consecuencia, autorizo su presentación para los fines legales pertinentes.

(f) ..

MSc. Fernando Valencia Aguirre

C.C.: 1003188669

DEDICATORIA

A mi familia, por ser mi pilar inquebrantable en cada etapa de mi vida. A mis padres, por su amor incondicional, sus sacrificios y su constante apoyo, sin los cuales este logro no habría sido posible. A mi hermanita, por ser mi inspiración y mi compañía en este camino. Gracias por enseñarme el valor del esfuerzo y la perseverancia.

A Dios, por darme la fortaleza, la sabiduría y la paciencia para superar cada obstáculo. Su guía ha iluminado mi camino y me ha dado la certeza de que, con fe y determinación, todo es posible.

A mí mismo, por la dedicación, la disciplina y la resiliencia para alcanzar esta meta. Por las noches de esfuerzo, las dudas superadas y la convicción de no rendirme. Este logro es el reflejo de todo el trabajo y la pasión que he puesto en mi crecimiento personal y profesional.

Con gratitud y orgullo, dedico este trabajo a quienes han sido mi mayor fuente de motivación.

AGRADECIMIENTO

En primer lugar, quiero expresar mi más profundo agradecimiento a mis padres, Rosa Iñiguez y Leoncio Lanchi, quienes han sido mi mayor inspiración y apoyo incondicional a lo largo de este camino. Su amor, sacrificio y motivación constante han sido fundamentales para que pudiera alcanzar esta meta. A mi hermanita Saraí, por ser una fuente inagotable de alegría y por recordarme siempre la importancia de seguir adelante con entusiasmo y determinación.

A mi Tutor, el Ingeniero Fernando Valencia, por su invaluable guía, paciencia y compromiso en cada etapa de este proyecto. Su conocimiento y orientación fueron clave para el desarrollo de esta investigación. Asimismo, agradezco al Ingeniero Marco Ciaccia, mi asesor, por sus aportes, su disposición y por compartir su experiencia, lo que permitió mejorar y fortalecer este trabajo.

A mi enamorada Catalina Ramos, por ser mi compañera en este proceso, por su apoyo incondicional, su confianza en mí y por motivarme a seguir adelante incluso en los momentos más difíciles. Su presencia ha sido un gran impulso en esta etapa de mi vida.

A mis queridos amigos, especialmente a Miguel Ángel Guerrero y a "Los Ratotas", quienes han estado a mi lado brindándome su apoyo, palabras de aliento y compañía a lo largo de este camino. Gracias por cada momento compartido, por las risas en los días difíciles y por recordarme la importancia de disfrutar cada paso del proceso. ¡Su amistad ha sido un pilar fundamental en este viaje!

A cada una de estas personas, mi más sincera gratitud, pues sin su apoyo y confianza, este logro no habría sido posible.

RESUMEN

El diseño de mecanismos de cuatro barras es fundamental en diversas aplicaciones de la ingeniería, como en la biomecánica para el desarrollo de prótesis de rodilla policéntricas. En este trabajo se desarrolló un programa computacional basado en un algoritmo evolutivo para optimizar el diseño de estos mecanismos, asegurando que la trayectoria del Centro Instantáneo de Rotación (CIR) se ajuste a las necesidades específicas de cada usuario.

Para ello, se implementó un algoritmo evolutivo en Python, utilizando bibliotecas como NumPy, DEAP y Tkinter, lo que permitió automatizar la búsqueda de los parámetros óptimos del mecanismo. Además, se incorporó una interfaz gráfica (GUI) que facilita la interacción con el usuario, permitiendo la entrada de datos personalizados, como el ángulo máximo de flexo-extensión y los parámetros de los eslabones.

El proceso de validación se realizó mediante la comparación de los resultados obtenidos con datos experimentales y modelos diseñados en SolidWorks, evaluando el error euclidiano para medir la precisión de la trayectoria generada. Los resultados mostraron que un mayor rango de movimiento mejora la precisión del mecanismo, reduciendo el error promedio de 17.83 unidades (para 20° de flexión) a 2.33 unidades (para 180° de flexión).

En conclusión, el programa desarrollado permite optimizar el diseño de mecanismos de cuatro barras de manera eficiente y adaptable a diferentes condiciones, ofreciendo una herramienta innovadora para el diseño.

Palabras clave: algoritmos evolutivos, mecanismo de cuatro barras, Centro Instantáneo de Rotación, Python, interfaz gráfica de usuario.

ABSTRACT

The design of four-bar mechanisms is fundamental in various engineering applications, such as biomechanics for the development of polycentric knee prostheses. In this work, a computational program based on an evolutionary algorithm was developed to optimize the design of these mechanisms, ensuring that the trajectory of the Instantaneous Center of Rotation (CIR) adapts to the specific needs of each user.

To achieve this, an evolutionary algorithm was implemented in Python, utilizing libraries such as NumPy, DEAP, and Tkinter, which allowed for the automation of the search for the mechanism's optimal parameters. Additionally, a graphical user interface (GUI) was incorporated to facilitate user interaction, allowing the input of customized data, such as the maximum flexion-extension angle and link parameters.

The validation process was carried out by comparing the obtained results with experimental data and models designed in SolidWorks, evaluating the Euclidean error to measure the accuracy of the generated trajectory. The results showed that a greater range of motion improves the mechanism's precision, reducing the average error from 17.83 units (for 20° flexion) to 2.33 units (for 180° flexion).

In conclusion, the developed program efficiently optimizes the design of four-bar mechanisms, making it adaptable to different conditions and providing an innovative tool for mechanism design.

Keywords: Evolutionary algorithms, four-bar mechanism, Instantaneous Center of Rotation, Python, graphical user interface.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO I: INTRODUCCIÓN

La incorporación del concepto de centro de rotación instantáneo (CIR) en el diseño de dispositivos médicos ha marcado un antes y un después en la biomecánica aplicada al cuidado de la salud. Al identificar y aprovechar este punto de rotación dinámico, estos sistemas logran una sincronización más precisa con el movimiento humano, optimizando tanto el desempeño como la experiencia del usuario. En el ámbito ortopédico, este principio ha sido clave para crear prótesis y órtesis altamente personalizadas. Por ejemplo, las prótesis de cadera modernas simulan de manera precisa la rotación natural de la articulación, gracias a la consideración del centro de rotación instantáneo. Este enfoque garantiza una mayor estabilidad y un rango de movimiento más fisiológico. Asimismo, la cirugía robótica y la terapia física se benefician del CIR para ofrecer tratamientos más precisos y seguros. En el caso de las prótesis de rodilla, el CIR dinámico permite adaptar la prótesis a cada paciente, mejorando su funcionalidad y confort.

La falta de congruencia entre un dispositivo médico y el movimiento natural de una articulación, específicamente su centro de rotación instantáneo, puede originar diversas complicaciones. Entre ellas se encuentran el desgaste acelerado del dispositivo, molestias e incluso dolor para el usuario debido a movimientos forzados, y una disminución en la eficacia del tratamiento. Estas consecuencias pueden llevar al rechazo del dispositivo por parte del paciente.

La creación de dispositivos médicos personalizados, adaptados a las características individuales de cada paciente, presenta un desafío tecnológico y económico significativo. La consideración de parámetros como el centro de rotación instantáneo y la anatomía específica exige el empleo de tecnologías avanzadas, como la impresión 3D y el modelado por computadora. Si bien esto permite una mayor precisión, eficacia y a la vez permite una mayor satisfacción del paciente y una mejor calidad de vida, también incrementa sustancialmente los costos de producción.

1.1 Planteamiento del problema

Actualmente la tecnología ha avanzado y evolucionado de manera considerable, principalmente en países de mayor desarrollo; ya sea porque cuentan con laboratorios de mejor calidad o por otros factores. Debido a esta situación en países en vías de desarrollo,

usualmente, se tiende a usar la tecnología proveniente del exterior, provocando que en muchas ocasiones no sean adaptables en el lugar de uso[1]

Tal es el caso del diseño de prótesis externas policéntricas o férulas de rodilla. Estos dispositivos están basados en mecanismos que cumplen una trayectoria de centro instantáneo de rotación (CIR) variable con respecto a los movimientos de flexo-extensión de la pierna. La totalidad de estas prótesis comercializadas en el Ecuador cumplen con una trayectoria proveniente de la antropometría europea [2].

Investigaciones recientes demuestran que el movimiento del centro de rotación instantáneo (CIR) es único para cada individuo. Por consiguiente, el uso de dispositivos médicos con movimientos preestablecidos puede generar complicaciones. Un software intuitivo que permita diseñar mecanismos de cuatro barras personalizados, ajustados a la trayectoria del CIR de cada paciente, sería fundamental para prevenir daños y mejorar la eficacia de los tratamientos.

1.2. Objetivos

1.2.1 Objetivo General

Desarrollar un programa para el diseño de mecanismos de 4 barras basado en un algoritmo evolutivo.

1.2.2 Objetivos Específicos

- Interpretar el basamento matemático para el desarrollo del programa computacional.
- Diseñar la interfaz humano-máquina.
- Implementar la interfaz y el algoritmo evolutivo.
- Validar el programa computacional.

1.3. Alcance

Proporcionar un programa computacional que permita la síntesis de mecanismos de 4 barras a través de un software libre.

Se busca contribuir con un diseño personalizado, es decir proponer un mecanismo adecuado de 4 barras con su centro instantáneo de rotación; donde, en este caso la facilidad principal que proponga sea reducir tiempos en la toma de decisiones y procesamiento de datos.

Se le brindará una interfaz humano máquina al algoritmo desarrollado por la Universidad técnica del Norte.

1.4. Justificación

La presente investigación está dirigida a la interpretación y al análisis de datos obtenidos de los cálculos y procesos que se siguen cuando se elaboran mecanismos que trabajan con 4 barras. Para facilitar este proceso, se emplea un algoritmo computacional que permite optimizar la elaboración y el diseño de estos mecanismos.

Desde una perspectiva económica, la implementación de esta solución tiene un impacto significativo, ya que reduce el tiempo necesario para diseñar mecanismos que satisfagan una trayectoria específica del Centro Instantáneo de Rotación. Al optimizar el proceso de diseño, se logra una mayor eficiencia en la producción, lo que conlleva ahorros en costos de mano de obra y en recursos materiales. Este enfoque también fomenta una mayor competitividad, ya que permite acortar los tiempos de desarrollo y responder de manera más ágil a las demandas del mercado, particularmente en sectores como la fabricación de dispositivos médicos y sistemas mecánicos avanzados [1].

En el ámbito académico, esta herramienta computacional ofrece importantes beneficios al facilitar la comprensión y el aprendizaje de los estudiantes en el diseño de mecanismos. Tradicionalmente, el diseño de mecanismos de 4 barras implica realizar cálculos manuales complejos y laboriosos. Sin embargo, con el uso de algoritmos específicos, los estudiantes pueden centrarse en comprender los principios fundamentales y la aplicación práctica, en lugar de invertir grandes cantidades de tiempo en procesos repetitivos. De igual manera, esta tecnología permitirá modelar y analizar una configuración específica de mecanismos, lo que resulta en una experiencia educativa más rica y eficiente [3].

Desde el punto de vista técnico, el método de síntesis de mecanismos propuesto mediante el uso de algoritmos no solo mejora la precisión en los resultados, sino que también proporciona herramientas para evaluar rápidamente distintas trayectorias posibles del CIR. Esto es particularmente relevante en aplicaciones donde la exactitud y la eficiencia son críticas, como en el diseño de prótesis de rodilla, brazos robóticos y dispositivos ortopédicos. El algoritmo también permite incorporar restricciones

específicas de diseño, como la optimización de materiales o la minimización de fricciones, haciendo que los mecanismos resultantes sean más funcionales y duraderos.

Un aspecto destacado de esta tecnología es el impacto positivo en los pacientes al permitir el diseño de prótesis personalizadas. Tener una prótesis diseñada específicamente para las necesidades y características físicas de cada individuo asegura un ajuste más preciso, una mayor comodidad y una mejor funcionalidad. Esto no solo mejora significativamente la calidad de vida de los pacientes, sino que también reduce el riesgo de complicaciones asociadas a dispositivos genéricos mal ajustados. Además, al replicar de manera más exacta los movimientos naturales del cuerpo, estas prótesis personalizadas permiten una recuperación más rápida y una integración más efectiva en las actividades diarias [4].

Por otro lado, la integración de esta tecnología en el campo industrial y educativo también contribuye al desarrollo sostenible, al reducir el desperdicio de materiales y optimizar los procesos de fabricación. En un entorno cada vez más orientado a la digitalización, estas herramientas se posicionan como una solución clave para enfrentar los retos del diseño mecánico moderno. En síntesis, esta investigación no solo busca facilitar la síntesis de mecanismos, sino también impulsar la innovación tecnológica en diferentes sectores, ahorrando tiempo, costos y recursos mientras se mejora la calidad y funcionalidad de los dispositivos diseñados.

CAPÍTULO II: MARCO REFERENCIAL

2.1 Antecedentes

En este proyecto, se llevó a cabo una investigación basada en la revisión de artículos científicos, tanto teóricos como de campo. El objetivo principal fue comprender de manera más profunda el tema de investigación propuesto y plantear una posible solución para abordar los obstáculos identificados.

Se evidenció en una investigación debidamente fundamentada que, para mejorar la adaptabilidad y funcionalidad de las prótesis de miembros inferiores, el rendimiento biomimético de la articulación de la rodilla es de suma importancia. En este contexto, se empleó el sistema de captura de movimiento óptico 3D Nokov para recopilar datos de movimiento de extremidades inferiores humanas normales, permitiendo la obtención del centro instantáneo de movimiento de la articulación de la rodilla en diferentes tipos de marcha [5].

Basándose en un algoritmo genético y tomando como función objetivo el error de la línea de centro instantáneo de movimiento de la rodilla, se diseñó un mecanismo de rodilla protésica de seis barras. Los resultados experimentales indican una similitud fundamental entre la trayectoria de movimiento del centro instantáneo de la articulación de la rodilla y la de una rodilla humana, lo que confiere a las prótesis una capacidad excepcional para replicar diversos tipos de marcha de manera biomimética. Este enfoque de adquisición de marcha no solo proporciona datos valiosos para los diseñadores de prótesis, sino que también promete una aplicación extendida en el diseño de prótesis y otros campos relacionados [5].

En este estudio, se propone un enfoque innovador para el diseño de mecanismos de 4 barras mediante el uso de algoritmos evolutivos. El enfoque se fundamenta en la implementación de operadores genéticos, como el cruce, la mutación y la selección, utilizando MATLAB. Se presenta un ejemplo ilustrativo detallado para demostrar la capacidad de los algoritmos evolutivos de encontrar soluciones óptimas globales o cercanas a globales de funciones multimodales [6].

Además, se muestra una aplicación de estos algoritmos en el diseño de un controlador robusto para sistemas de control inciertos, evidenciando su potencial en el desarrollo de sistemas inteligentes de ingeniería. Este trabajo aporta una herramienta

novedosa y eficaz para el diseño de mecanismos de 4 barras, mejorando así la eficiencia y la precisión en este campo de estudio [6].

Por otra parte se desarrolló un trabajo de investigación que proporciona una introducción breve al algoritmo genético, seguida de una explicación detallada de su implementación utilizando MATLAB. Además, se demuestra su utilidad en la optimización de funciones y la resolución de ecuaciones mediante tres ejemplos prácticos. Se aborda también la estrategia para evitar la optimización local mediante la modificación de parámetros clave. Este enfoque promete contribuir significativamente a la investigación sobre uso del software que se va a usar [7].

Adicionalmente, existe un trabajo que presenta un programa para diseñar mecanismos de 4 barras basado en el algoritmo evolutivo BIANCA 3.1. Destaca por su capacidad de resolver problemas de optimización en ingeniería, manejar restricciones, y su flexibilidad en la interfaz entre el código del usuario y métodos iterativos. La versión 3.1 va más allá del estándar del GA al permitir la evolución simultánea de individuos y especies, siendo crucial para optimizar sistemas modulares. Esto proporciona una herramienta avanzada para el diseño eficiente de mecanismos de 4 barras en Python [8].

De igual forma, Darrell Whitley presenta un programa innovador para el diseño de mecanismos de 4 barras, basado en un algoritmo evolutivo. Se abordan conceptos clave del algoritmo genético, incluyendo formas experimentales como modelos paralelos de islas y algoritmos genéticos celulares paralelos. Además, se explora la búsqueda genética mediante muestreo de hiperplanos. Se revisan fundamentos teóricos esenciales, como el teorema del esquema y modelos exactos recientes del algoritmo genético canónico. Esta fuente proporciona información avanzada para el diseño eficiente del mecanismo, integrando aspectos prácticos de los algoritmos genéticos [9].

Este proyecto aborda una necesidad fundamental en el diseño de mecanismos de 4 barras al superar las limitaciones de los métodos clásicos de síntesis. Dichos métodos enfrentan restricciones en la definición precisa de trayectorias debido a su enfoque gráfico y analítico. En contraste, el software desarrollado emplea Algoritmos Genéticos para optimizar la generación de trayectorias y la simulación cinemática, permitiendo la consideración de múltiples puntos de precisión. Esto ofrece una herramienta avanzada y eficiente para los ingenieros, al simplificar y acelerar el proceso de síntesis de

mecanismos, mientras que las gráficas generadas facilitan la interpretación de los datos y validan la precisión del software [10].

En este proyecto de titulación, se destaca la relevancia de la implementación de algoritmos genéticos en Python para el diseño de mecanismos de 4 barras. Los algoritmos genéticos son una herramienta poderosa basada en la selección natural y la genética, permitiendo una búsqueda probabilística eficiente de soluciones. La elección de este entorno se justifica por su capacidad para procesar listas o cadenas de manera más productiva que otros lenguajes como C/C++/Java. Además, la implementación de algoritmos genéticos en este entorno es rápida y sencilla, lo que agiliza el desarrollo del programa para el diseño de mecanismos. Este enfoque proporciona una base sólida y eficiente para el desarrollo del programa, asegurando su viabilidad y efectividad en la búsqueda de soluciones óptimas para el diseño de mecanismos de 4 barras [11].

Además, existe un artículo en el que se enfatiza la importancia de la implementación de algoritmos genéticos modernos y paralelos en Python para el diseño de mecanismos de 4 barras. Estos algoritmos, derivados de principios naturales y fenómenos evolutivos, son intrínsecamente adecuados para el procesamiento paralelo, lo que resulta en una mayor velocidad y optimización del proceso. El documento aborda específicamente formas seleccionadas de paralelización de algoritmos genéticos, mostrando su aplicación práctica mediante la implementación en este entorno. Se exploran cuatro módulos que ofrecen procesamiento paralelo, lo que permite una distribución eficiente de la carga y una significativa mejora en la velocidad del algoritmo [12].

Otra investigación reconoce la importancia del basamento matemático proporcionado por la ecuación de Freudenstein en el diseño de mecanismos de 4 barras. Es reconocido como el padre de la cinemática moderna de mecanismos y máquinas, y su trabajo ha tenido un impacto significativo en la investigación académica e industrial, así como en la enseñanza y práctica relacionada con el análisis y diseño de mecanismos en todo el mundo. La ecuación de Freudenstein, derivada de un enfoque analítico, es fundamental para comprender y diseñar mecanismos de cuatro barras, que son comunes en numerosas máquinas utilizadas en la vida cotidiana. Su estudio y aplicación en este trabajo proporcionan una sólida base teórica para el desarrollo del programa de diseño de

mecanismos de 4 barras basado en un algoritmo evolutivo, asegurando la precisión y eficacia del mismo [13].

Adicionalmente, se profundiza en el análisis numérico de mecanismos mediante el estudio de la ecuación de Freudenstein y su aplicación en el diseño de la falange proximal de un dedo antropomórfico. Se abordan los desafíos numéricos inherentes a esta ecuación, evaluando su consistencia y condicionamiento con el algoritmo de SOR, o sobre-relajación sucesiva (Successive Over-Relaxation en inglés). El enfoque heurístico utilizado busca mejorar la convergencia y precisión del sistema. La implementación de este basamento matemático en el programa para el diseño de mecanismos de 4 barras basado en un algoritmo evolutivo proporciona una sólida fundamentación teórica y práctica para la optimización y análisis de estos sistemas [14].

2.2 Marco Teórico

Las bases teóricas se fundamentan en la aplicación de algoritmos genéticos y la ecuación de Freudenstein en el diseño de mecanismos de 4 barras. Los algoritmos genéticos, inspirados en la evolución natural, ofrecen una metodología eficiente para la optimización y síntesis de mecanismos, permitiendo explorar múltiples soluciones y encontrar óptimos globales. La ecuación de Freudenstein, desarrollada por Ferdinand Freudenstein, constituye un pilar fundamental en la cinemática de mecanismos y máquinas, brindando un enfoque analítico para el diseño de mecanismos de cuatro barras comúnmente utilizados en diversas aplicaciones industriales y cotidianas como lo son sistemas de suspensión automotriz, brazos manipuladores, puertas articuladas, entre otros [15].

Además, se aprovecha el conocimiento derivado de la investigación de Freudenstein y sus colaboradores para abordar problemas numéricos en el diseño de mecanismos, como la implementación del algoritmo genético para mejorar la convergencia y precisión de los sistemas. Esta combinación de bases teóricas proporciona una sólida fundamentación para el desarrollo del programa de diseño de mecanismos de 4 barras, asegurando su eficacia y precisión en la búsqueda de soluciones óptimas para aplicaciones en ingeniería mecatrónica [3]**.**

2.2.1 Uso de Python como software libre

En el ámbito de la ingeniería, Python se destaca como una herramienta versátil y poderosa, ampliamente utilizada en el desarrollo de software para diversas aplicaciones. Este lenguaje es de alto nivel, lo que significa que su sintaxis es fácil de leer y entender, resultando ideal para ingenieros y científicos que desean desarrollar aplicaciones complejas. Una de sus características más destacadas es la amplia gama de bibliotecas y herramientas disponibles, que permiten a los desarrolladores implementar algoritmos complejos de manera eficiente y efectiva. En particular, es utilizado frecuentemente en el desarrollo de algoritmos genéticos, una técnica de optimización inspirada en la evolución natural. Los métodos evolutivos funcionan mediante la generación de una población inicial de soluciones aleatorias, que luego se someten a procesos de selección, cruzamiento y mutación para producir nuevas generaciones de soluciones que se acercan cada vez más a una solución óptima. En este lenguaje, los métodos evolutivos se pueden implementar utilizando bibliotecas como NumPy, SciPy y DEAP, que proporcionan herramientas específicas para la creación y ejecución de estos algoritmos [16]

En cuanto a la creación de una interfaz de usuario para facilitar el uso práctico del algoritmo de diseño de mecanismos de 4 barras basado en un algoritmo evolutivo, este lenguaje de programación ofrece una amplia gama de herramientas y bibliotecas para el desarrollo de interfaces gráficas de usuario (GUI). Una de las bibliotecas más populares para crear GUI es Tkinter, que proporciona un conjunto de widgets y herramientas para desarrollar interfaces gráficas interactivas de manera rápida y sencilla. Con Tkinter, los desarrolladores pueden crear ventanas, botones, cuadros de texto y otros elementos de interfaz que permiten a los usuarios interactuar con el programa de manera intuitiva y eficiente. Además, este lenguaje también es compatible con otras bibliotecas de GUI, como PyQt y wxPython, que ofrecen características adicionales y flexibilidad para el desarrollo de interfaces más complejas. En resumen, este entorno de desarrollo proporciona una solución completa y poderosa para la creación de interfaces de usuario intuitivas y prácticas que facilitan el uso del algoritmo de diseño de mecanismos de 4 barras en la práctica ingenieril [17].

2.2.2 Uso de Ruby como software libre

En el ámbito de la ingeniería, Ruby se destaca como una herramienta poderosa y flexible, ampliamente utilizada en el desarrollo de software para diversas aplicaciones.

Este lenguaje de alto nivel es conocido por su sintaxis limpia y legible, lo que lo hace ideal para ingenieros y científicos que desean desarrollar aplicaciones complejas de manera eficiente. Una de las características más notables de Ruby es su amplio ecosistema de bibliotecas y herramientas, que permite a los desarrolladores implementar algoritmos avanzados con facilidad.

En particular, Ruby se utiliza frecuentemente en el desarrollo de algoritmos genéticos, una técnica de optimización inspirada en la evolución natural. Los algoritmos genéticos en Ruby funcionan generando una población inicial de soluciones aleatorias, que luego se someten a procesos de selección, cruzamiento y mutación para producir nuevas generaciones de soluciones que se aproximan cada vez más a una solución óptima. Estos métodos evolutivos se pueden implementar utilizando bibliotecas como 'genetic', que proporciona herramientas específicas para la creación y ejecución de algoritmos evolutivos. La simplicidad de Ruby y su capacidad para manejar algoritmos complejos lo convierten en una opción atractiva para los proyectos de optimización en ingeniería [18].

En el contexto de la creación de una interfaz de usuario para facilitar el uso práctico del algoritmo de diseño de mecanismos de 4 barras basado en un algoritmo evolutivo, Ruby ofrece una variedad de herramientas y bibliotecas para el desarrollo de interfaces gráficas de usuario (GUI). Una de las bibliotecas más populares para crear GUI en Ruby es Shoes. Shoes proporciona un conjunto de widgets y herramientas que permiten a los desarrolladores construir interfaces gráficas interactivas de manera rápida y sencilla. Con Shoes, es posible crear ventanas, botones, cuadros de texto y otros elementos de interfaz que permiten a los usuarios interactuar con el programa de manera intuitiva y eficiente.

Además, Ruby es compatible con otras bibliotecas de GUI como Tk y FXRuby, que ofrecen características adicionales y flexibilidad para el desarrollo de interfaces más complejas. Tk es conocida por su simplicidad y rápida integración con Ruby, lo que la hace ideal para aplicaciones básicas. FXRuby, por otro lado, ofrece una mayor flexibilidad y opciones avanzadas para diseñar interfaces gráficas más sofisticadas y estéticamente agradables. Estas herramientas permiten a los desarrolladores de Ruby crear interfaces gráficas de usuario de forma eficiente, mejorando la accesibilidad y la experiencia del usuario final [18].

En resumen, Ruby proporciona un entorno de desarrollo completo y robusto para la creación de interfaces de usuario intuitivas y prácticas, facilitando el uso del algoritmo de diseño de mecanismos de 4 barras en la práctica ingenieril. Su capacidad para manejar algoritmos complejos, combinada con su facilidad para desarrollar interfaces gráficas de usuario, hace de Ruby una opción ideal para proyectos en ingeniería mecatrónica que requieren tanto robustez en los cálculos como facilidad en la interacción con el usuario. Esto lo convierte en una herramienta valiosa para los ingenieros que buscan optimizar y mejorar sus procesos de diseño y análisis.

2.2.3 Uso de Perl como software libre

En el ámbito de la ingeniería, Perl se destaca como una herramienta versátil y poderosa, ampliamente utilizada en el desarrollo de software para diversas aplicaciones. Este lenguaje es de alto nivel, lo que significa que su sintaxis es flexible y adaptable, resultando ideal para ingenieros y científicos que desean desarrollar aplicaciones complejas. Una de sus características más destacadas es la vasta colección de módulos y herramientas disponibles, que permiten a los desarrolladores implementar algoritmos complejos de manera eficiente y efectiva. En particular, es utilizado frecuentemente en el desarrollo de algoritmos genéticos, una técnica de optimización inspirada en la evolución natural. Los métodos evolutivos funcionan mediante la generación de una población inicial de soluciones aleatorias, que luego se someten a procesos de selección, cruzamiento y mutación para producir nuevas generaciones de soluciones que se acercan cada vez más a una solución óptima. En Perl, los métodos evolutivos se pueden implementar utilizando módulos como Algorithm::Evolutionary, que proporciona herramientas específicas para la creación y ejecución de estos algoritmos [19].

En cuanto a la creación de una interfaz de usuario para facilitar el uso práctico del algoritmo de diseño de mecanismos de 4 barras basado en un algoritmo evolutivo, Perl ofrece una amplia gama de herramientas y bibliotecas para el desarrollo de interfaces gráficas de usuario (GUI). Una de las bibliotecas más populares para crear GUI en Perl es Tk, que proporciona un conjunto de widgets y herramientas para desarrollar interfaces gráficas interactivas de manera rápida y sencilla. Con Tk, los desarrolladores pueden crear ventanas, botones, cuadros de texto y otros elementos de interfaz que permiten a los usuarios interactuar con el programa de manera intuitiva y eficiente. Además, Perl también es compatible con otras bibliotecas de GUI, como WxPerl y Gtk2-Perl, que

ofrecen características adicionales y flexibilidad para el desarrollo de interfaces más complejas [19].

En resumen, este entorno de desarrollo proporciona una solución completa y poderosa para la creación de interfaces de usuario intuitivas y prácticas que facilitan el uso del algoritmo de diseño de mecanismos de 4 barras en la práctica ingenieril. La capacidad de Perl para manejar algoritmos complejos y desarrollar interfaces gráficas de usuario eficientes hace de él una opción ideal para proyectos en ingeniería mecatrónica que requieren robustez en los cálculos y facilidad en la interacción con el usuario

2.2.4 Selección del posible software libre de programación que facilite el trabajo con algoritmos genéticos.

Con el propósito de seleccionar la opción más adecuada para el proyecto, se realizaron evaluaciones exhaustivas de diversos lenguajes de programación. Los criterios de evaluación incluyeron la capacidad de trabajo, la disponibilidad de funciones apropiadas, la facilidad de uso, y la condición de que el lenguaje debe ser libre. Se examinó cómo cada lenguaje se adapta a estos requerimientos, considerando que la libertad de uso es crucial para evitar restricciones de licencia y costos adicionales. A continuación, se presenta el lenguaje de programación que mejor cumple con estos criterios, destacando que ofrecen un equilibrio óptimo entre funcionalidad, accesibilidad y licencia libre.

En el desarrollo de esta investigación, se ha optado por utilizar Python como el lenguaje principal de programación. Python ofrece numerosas ventajas para el manejo de algoritmos genéticos en el contexto de este estudio. Su sintaxis clara y legible facilita la implementación de algoritmos, mientras que su amplia gama de bibliotecas especializadas simplifica el desarrollo y la experimentación. Bibliotecas como NumPy y SciPy proporcionan herramientas avanzadas para la manipulación de datos y la ejecución de operaciones matemáticas, mientras que DEAP (Distributed Evolutionary Algorithms in Python) es particularmente útil para implementar y optimizar algoritmos evolutivos.

La flexibilidad de Python permite ajustar parámetros y ejecutar pruebas de manera eficiente, lo que acelera el proceso de desarrollo y optimización. Además, Python es conocido por su capacidad para integrar fácilmente diversos módulos y herramientas,

proporcionando un entorno de desarrollo altamente versátil. La comunidad activa y extensa de Python también ofrece acceso a numerosos recursos y soporte técnico, lo que facilita la resolución de problemas y la mejora continua del código.

Además, Python se distingue por su habilidad para crear interfaces gráficas de usuario que son tanto simples como efectivas. Utilizando bibliotecas como Tkinter y PyQt, los programadores pueden diseñar interfaces que permiten una interacción clara y fluida con el software. Tkinter es especialmente apreciado por su facilidad de uso y rápida integración con Python, siendo ideal para la construcción de aplicaciones básicas y funcionales. En contraste, PyQt proporciona una mayor flexibilidad y una gama de opciones avanzadas para desarrollar interfaces gráficas más elaboradas y estéticamente sofisticadas. Esta capacidad de Python para desarrollar interfaces gráficas de manera eficiente contribuye a mejorar tanto la accesibilidad como la experiencia del usuario. Al combinar un entorno robusto para cálculos con la facilidad de crear interfaces amigables, Python se posiciona como una opción excelente para el desarrollo de aplicaciones que necesitan ofrecer tanto potencia en el procesamiento de datos como facilidad en la interacción con el usuario.

Por último, Python es un lenguaje de código abierto, lo que asegura que los desarrolladores tengan acceso a sus características sin costo adicional, contribuyendo a la accesibilidad y a la adaptabilidad del lenguaje para proyectos de investigación. Esto convierte a Python en una opción ideal para llevar a cabo esta investigación, respaldada por una plataforma confiable y ampliamente soportada.

2.2.5 Uso de la ecuación de Freudenstein en el desarrollo del mecanismo de 4 barras

En el campo de la ingeniería mecánica, la ecuación de Freudenstein es crucial para el diseño y análisis de mecanismos de 4 barras, particularmente en el desarrollo de prótesis de rodilla policéntrica. Creada por Ferdinand Freudenstein, esta ecuación es una herramienta esencial para entender y modelar el movimiento de estos sistemas mecánicos [2]. Los mecanismos de 4 barras son estructuras articuladas compuestas por cuatro elementos conectados por articulaciones, que permiten la conversión de movimiento rotacional en movimiento lineal y viceversa. En el diseño de prótesis de rodilla policéntrica, los mecanismos de 4 barras son utilizados para replicar el movimiento

natural de la rodilla humana, permitiendo a los usuarios realizar actividades cotidianas con mayor comodidad y naturalidad.

La aplicación de la ecuación de Freudenstein en el diseño de mecanismos de 4 barras para prótesis de rodilla policéntrica implica la determinación precisa de los parámetros geométricos y cinemáticos del sistema. Estos parámetros incluyen la longitud de los eslabones, las ubicaciones de las articulaciones y las restricciones de movimiento, que se utilizan para calcular el movimiento del mecanismo en respuesta a las fuerzas aplicadas. La ecuación de Freudenstein permite modelar y analizar el comportamiento cinemático del mecanismo, lo que facilita el diseño de prótesis de rodilla que se ajusten de manera óptima a las necesidades individuales de los usuarios. Además, la aplicación de esta ecuación en el diseño de prótesis de rodilla policéntrica permite mejorar la estabilidad, la funcionalidad y la comodidad de las prótesis, lo que contribuye a mejorar la calidad de vida de las personas con discapacidades físicas.

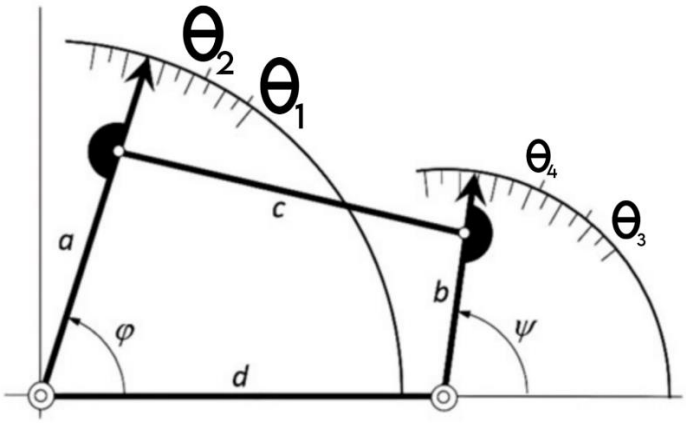
En cuanto a la implementación práctica de la ecuación de Freudenstein y los mecanismos de 4 barras en el desarrollo del programa para el diseño de mecanismos de 4 barras basado en un algoritmo evolutivo, Python se presenta como una herramienta versátil y poderosa. Python ofrece una amplia gama de bibliotecas y herramientas para el desarrollo de algoritmos genéticos y la creación de interfaces de usuario interactivas. Mediante el uso de bibliotecas como NumPy, SciPy y DEAP, los desarrolladores pueden implementar algoritmos genéticos eficientes para la optimización de los parámetros de los mecanismos de 4 barras. Además, Python ofrece bibliotecas como Tkinter, PyQt y wxPython para la creación de interfaces gráficas de usuario intuitivas y fáciles de usar, lo que facilita la interacción con el programa y la visualización de los resultados. En resumen, la combinación de la ecuación de Freudenstein, los mecanismos de 4 barras y los algoritmos genéticos con Python proporciona una poderosa herramienta para el diseño y análisis de prótesis de rodilla policéntrica, que puede mejorar significativamente la calidad de vida de las personas con discapacidades físicas [16].

A continuación, se presenta la Figura 1 que muestra el funcionamiento del mecanismo basado en la ecuación de Freudenstein, utilizada para la síntesis y análisis de mecanismos de cuatro barras. En esta representación gráfica, se destacan las variables clave del sistema, como las longitudes de los eslabones, los ángulos de rotación de los distintos eslabones y la posición del Centro Instantáneo de Rotación. Además, se incluyen

las ecuaciones que rigen el comportamiento del mecanismo, proporcionando una base matemática para su análisis y optimización. La Figura 1 permite visualizar cómo interactúan estos parámetros en el movimiento del mecanismo, facilitando la comprensión de su comportamiento y su aplicación en el diseño de prótesis de rodilla policéntricas.

Figura 1.

Mecanismo de 4 Barras [10]



a, b, c y d: Son las líneas rectas que conectan los cuatro puntos de pivote del mecanismo. Son cada una de las barras que compone el mecanismo

Θ1, Θ2, Θ3 y Θ4: Representan las posiciones angulares de las barras en un instante dado. Estos ángulos son variables y cambian a medida que el mecanismo se mueve.

$$A = \frac{a^2 + b^2 - c^2 + d^2}{2(ab)} + \frac{d}{b} * cos\theta_1 - \frac{d}{a} * (cos\theta_1 * cos\theta_3 + sin\theta_1 * sin\theta_3) - cos\theta_3 \quad (1)$$

$$B = \frac{2 * (sin\theta_3 - d * sin\theta_1)}{b} \quad (2)$$

$$C = \frac{(a^2+b^2 - c^2 + d^2)}{(2ab)} - \frac{d(cos\theta_1 * cos\theta_3 + sin\theta_1 * sin\theta_3)}{a} + cos\theta_3 - \frac{d * cos\theta_1}{b} \quad (3)$$

$$\theta_4 = 2tan^{-1}\left[\frac{-E \pm \sqrt{E^2 - 4DF}}{2D}\right] \quad (4)$$

$$D = \frac{(b^2 - a^2 + c^2 + d^2)}{(2bc)} - \frac{d(cos\theta_1 * cos\theta_3 + sin\theta_1 * sin\theta_3)}{c} - \frac{d * cos\theta_1}{b} + cos\theta_3 \quad (5)$$

$$E = 2 * \left(\frac{d \times sin\theta_1}{b} - sin\theta_3\right) \quad (6)$$

$$F = \frac{(b^2 - a^2 + c^2 + d^2)}{(2bc)} - \frac{d(cos\theta_1 * cos\theta_3 + sin\theta_1 * sin\theta_3)}{c} + \frac{d * cos\theta_1}{b} - cos\theta_3 \quad (7)$$

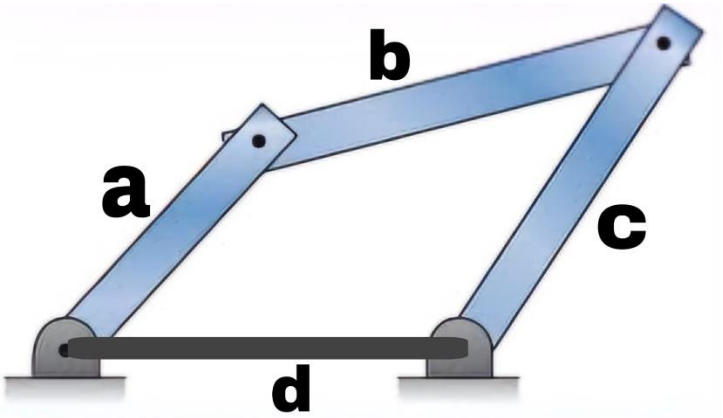
Las ecuaciones (1), (2), (3), (4), (5), (6) y (7) servirán como base fundamental para el cálculo y ejecución del algoritmo, ya que permiten modelar matemáticamente el comportamiento del mecanismo de cuatro barras. A través de estas ecuaciones, se pueden determinar las posiciones, ángulos y desplazamientos de los eslabones en función de las condiciones iniciales y los parámetros del sistema.

2.2.6 Tipos de mecanismos para articulaciones de rodillas

El mecanismo utilizado en la articulación de rodilla es un mecanismo de cuatro barras, como se muestra en la Figura 2. Para su análisis, se emplea un método analítico basado en la ecuación de Freudenstein. Esta ecuación es fundamental para la síntesis de mecanismos de cuatro barras, ya que permite determinar las dimensiones de los eslabones en un cuadrado articulado. Con esta herramienta, se establecen las relaciones entre las longitudes de los eslabones y los ángulos formados entre ellos, con el objetivo principal de simular el movimiento de la rodilla de manera precisa.

Figura 2.

Mecanismo de 4 barras modificado [20].



En el caso de las personas con amputación transfemoral, la estructura de la pierna se corta, dejando un muñón o miembro residual con solo una parte de la musculatura disponible. Debido a esta limitación, se han desarrollado prótesis externas que incluyen mecanismos monocéntricos y policéntricos para reemplazar la articulación de la rodilla. En una rodilla monocéntrica, el movimiento de flexión y extensión se realiza alrededor de un único eje fijo. Por otro lado, en una rodilla policéntrica, el eje de la articulación se desplaza en función del ángulo de flexión y extensión de la rodilla, lo que resulta en un Centro Instantáneo de Rotación.

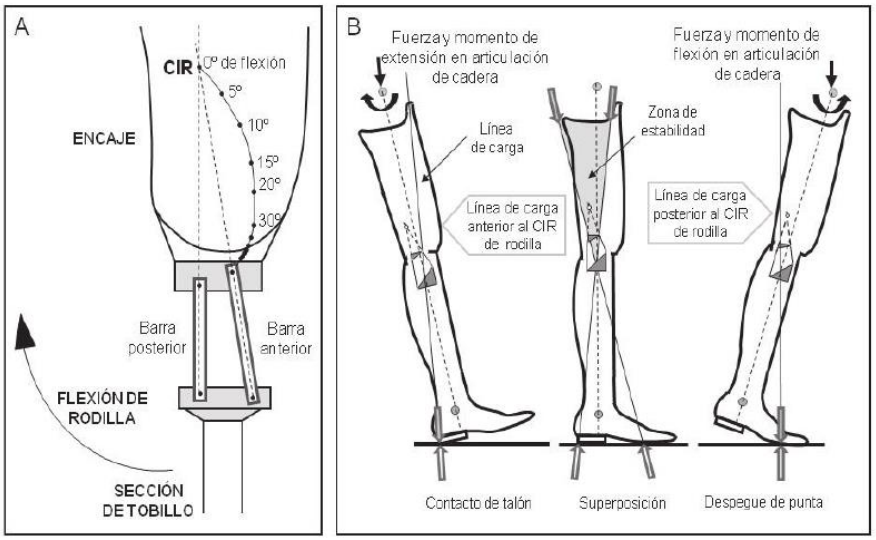
Para explicar mejor, en un mecanismo policéntrico de 4 barras, como se muestra en la Figura 2, el CIR se sitúa en la región de extensión entre las barras posterior y anterior. Este diseño permite al mecanismo ejecutar movimientos simultáneos de rotación y traslación durante la flexión de la rodilla. Esta capacidad de mover el eje de rotación es esencial para imitar de manera más natural el movimiento humano, proporcionando una funcionalidad más cercana a la de una rodilla biológica [10].

El uso de mecanismos policéntricos en prótesis de rodilla es beneficioso porque ofrecen una mejor estabilidad y un patrón de marcha más natural comparado con los mecanismos monocéntricos. En particular, los mecanismos de 4 barras policéntricos pueden mejorar la seguridad del usuario al reducir el riesgo de colapso inesperado de la rodilla, ya que el eje de rotación se posiciona de manera que ofrece una mayor resistencia a la flexión no deseada. Además, la variación del CIR permite una mejor adaptación a las diferentes fases del ciclo de la marcha, proporcionando un movimiento más suave y eficiente.

La Figura 3 muestra cómo el CIR se desplaza durante la extensión y flexión, resaltando la capacidad del mecanismo de 4 barras para realizar movimientos combinados de rotación y traslación. Este tipo de diseño es especialmente útil en prótesis de rodilla para amputados transfemorales, ya que puede compensar parcialmente la pérdida de la musculatura y ofrecer una funcionalidad más cercana a la natural [10].

Figura 3.

Movimiento de Flexión – Extensión [21].



2.2.7 Método de recolección de puntos en el espacio

Para obtener las coordenadas del Centro Instantáneo de Rotación de la rodilla de una persona, se pueden emplear varios métodos. Estos métodos varían en complejidad y precisión, y la elección del método puede depender de los recursos disponibles y del contexto específico de la medición. A continuación, se presenta algunos de los métodos más comunes:

2.2.7.1 Métodos Directos con Equipos de Medición.

La medición directa, mediante instrumentos especializados, es una herramienta indispensable en la ingeniería y la industria. Esta técnica permite obtener datos precisos y confiables de manera rápida y sencilla, facilitando el control de calidad y la optimización de procesos. Desde mecanismos para medir dimensiones hasta sensores de presión para evaluar fuerzas, estos dispositivos ofrecen una amplia gama de aplicaciones, desde la fabricación hasta la investigación

2.2.7.1.1 Goniometría electromagnética

La goniometría electromagnética es una técnica sofisticada que utiliza sensores electromagnéticos para registrar los ángulos y movimientos de las articulaciones en

tiempo real como se muestra en la Figura 4. Este enfoque se basa en la colocación de sensores especializados en puntos clave del cuerpo, los cuales detectan la posición y el movimiento de los segmentos corporales mediante la generación de campos electromagnéticos. Los sensores, que incluyen bobinas y transmisores, emiten y reciben señales electromagnéticas, lo que permite capturar de manera precisa y continua los movimientos de las articulaciones [4].

Figura 4.

Medición con Goniometría electromagnética [22]

En este método, los sensores se sitúan en distintas partes del cuerpo para medir ángulos de flexión y extensión, así como otros parámetros cinemáticos relacionados. La información recolectada por estos dispositivos es esencial para calcular el Centro Instantáneo de Rotación de la articulación estudiada. La principal ventaja de esta técnica es su habilidad para ofrecer datos en tiempo real con alta precisión, lo que posibilita un análisis exhaustivo y dinámico de los movimientos articulares [23].

Una vez obtenidos los datos, se procesan con software especializado que convierte las lecturas de los sensores en valores de ángulos y trayectorias de movimiento. Estos valores son fundamentales para determinar el CIR, ya que permiten observar cómo se comporta la articulación en diferentes rangos de movimiento. La capacidad de medir y analizar continuamente estos movimientos facilita una comprensión más profunda de la mecánica articular, lo que resulta particularmente útil en campos como la biomecánica y el diseño de prótesis o dispositivos ortopédicos.

En conclusión, la goniometría electromagnética ofrece un método preciso y eficaz para medir los movimientos articulares y calcular el CIR, jugando un papel crucial en la investigación y el desarrollo en biomecánica y rehabilitación [23].

2.2.7.1.2 Cinemática de marcadores

La técnica de cinemática de marcadores, conocida también como motion capture, es un método sofisticado que utiliza cámaras y marcadores para monitorizar y analizar los movimientos de las articulaciones del cuerpo como se indica en la Figura 5. En este procedimiento, se colocan marcadores en ubicaciones específicas del cuerpo del sujeto, y estos marcadores son seguidos por cámaras especializadas que capturan su posición en un espacio tridimensional [21].

Figura 5.

Medición con Cinemática de marcadores [21].

Las cámaras empleadas en este proceso están diseñadas para detectar la posición de los marcadores con gran precisión, lo que permite una evaluación minuciosa de los movimientos articulares. Cada marcador colocado en el cuerpo emite señales o refleja la luz, que son captadas por las cámaras desde diversos ángulos. Este sistema de captura se realiza a alta velocidad, proporcionando un registro continuo y exacto de los movimientos corporales en tiempo real.

Una vez que los datos de los marcadores son obtenidos, se procesan a través de software especializado en análisis de movimiento. Este software convierte las coordenadas espaciales de los marcadores en datos cinemáticos, tales como ángulos y trayectorias de movimiento. Con estos datos, el software puede calcular el Centro

Instantáneo de Rotación de las articulaciones estudiadas. El CIR se determina evaluando cómo se mueven los marcadores durante los movimientos articulares y cómo estos se relacionan con la rotación en torno a un punto específico [21].

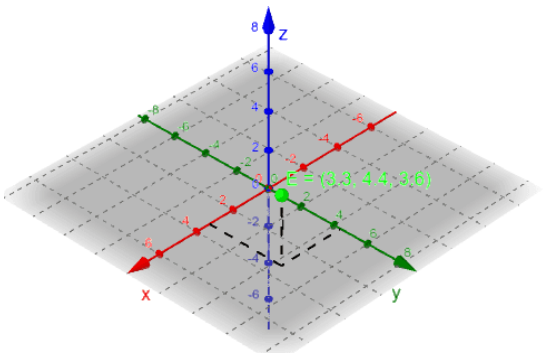
La principal ventaja de la cinemática de marcadores radica en su capacidad para ofrecer una visualización precisa y detallada de los movimientos articulares en tres dimensiones. Esto permite a investigadores y profesionales llevar a cabo un análisis exhaustivo del comportamiento de las articulaciones en diferentes actividades y rangos de movimiento. Esta técnica es especialmente útil en campos como la biomecánica, el diseño de prótesis y la rehabilitación, donde la precisión en el registro de movimientos es esencial para el desarrollo y ajuste de dispositivos y tratamientos. En definitiva, la cinemática de marcadores es una herramienta potente para investigar y comprender los movimientos articulares y calcular el CIR con gran exactitud [4].

2.2.7.1.3 Imágenes tridimensionales

De igual manera, el análisis de imágenes tridimensionales es una técnica avanzada que utiliza imágenes obtenidas a través de métodos de diagnóstico como la resonancia magnética (RM) y colocando su trayectoria en programas como se muestra en la Figura 6, para después generar un modelo 3D detallado de la rodilla. Este enfoque se basa en la captura de imágenes de alta resolución que muestran cortes transversales del cuerpo en varios planos. Estas imágenes se procesan y se combinan para formar un modelo tridimensional completo que representa tanto la estructura ósea como los tejidos blandos de la rodilla [7].

Figura 6.

GeoGebra en creación de puntos tridimensioanles



El proceso comienza con la obtención de múltiples imágenes en secciones transversales mediante RM, cada una correspondiendo a una capa específica del cuerpo. Mediante software especializado, estas imágenes se fusionan para crear un modelo 3D continuo que refleja con precisión la anatomía de la rodilla. Este modelo permite una visualización detallada de la articulación, abarcando huesos, cartílagos y ligamentos [23].

Una vez que se ha generado el modelo 3D de la rodilla, se puede utilizar para calcular el Centro Instantáneo de Rotación de la articulación. Este punto es fundamental para comprender el movimiento de la rodilla durante diferentes actividades. Para determinar el CIR, se analizan las posiciones y movimientos de las estructuras óseas y articulares en el modelo tridimensional, observando cómo cambian a lo largo del rango de movimiento [23].

El análisis de imágenes 3D proporciona una precisión excepcional en el estudio de la biomecánica de la rodilla al ofrecer una visión detallada de la anatomía y permitir una evaluación exacta del comportamiento articular. Esta técnica resulta particularmente valiosa en el ámbito clínico y en la investigación, como en el diseño de prótesis y la planificación quirúrgica, donde una comprensión precisa del CIR y de la mecánica de la rodilla es crucial para desarrollar soluciones personalizadas y efectivas. En resumen, el análisis de imágenes 3D facilita una evaluación completa y precisa del Centro Instantáneo de Rotación y de la funcionalidad de la rodilla.

2.2.7.2 Métodos Indirectos Basados en Modelos

Los métodos de medición indirectos, basados en modelos matemáticos y físicos, permiten estimar variables que son difíciles de medir directamente. Al combinar datos experimentales con simulaciones numéricas, estos métodos ofrecen una herramienta poderosa para analizar sistemas complejos en diversas disciplinas, desde la ingeniería hasta las ciencias ambientales. Su flexibilidad y capacidad para inferir información oculta los convierten en una herramienta esencial en la investigación y el desarrollo.

2.2.7.2.1 Modelos Biomecánicos

Los modelos biomecánicos juegan un papel crucial en el análisis del movimiento de la rodilla, ya que emplean representaciones matemáticas y biomecánicas para describir su funcionamiento. Estos modelos se fundamentan en la representación matemática de las estructuras y dinámicas articulares, teniendo en cuenta factores como las dimensiones

y los movimientos previstos. Utilizar estos modelos permite calcular el Centro Instantáneo de Rotación de la rodilla al examinar cómo interactúan los distintos componentes de la articulación durante el movimiento [16].

El proceso inicia con la creación de modelos matemáticos que representan las fuerzas, momentos y relaciones cinemáticas implicadas en el movimiento de la rodilla. Estos modelos integran parámetros como las longitudes óseas, la configuración de los ligamentos y el rango de movimiento de la articulación. A partir de estos parámetros, los modelos matemáticos permiten simular cómo se comporta la rodilla en diversas posiciones y movimientos [23].

Una vez formulados estos modelos, se aplican técnicas analíticas para calcular el CIR. Este cálculo se basa en la evaluación de las variaciones en la posición de la articulación durante la flexión y extensión, y en la relación con el comportamiento dinámico de la rodilla. Se utilizan ecuaciones biomecánicas para describir el movimiento y las fuerzas involucradas, lo que permite una estimación precisa del CIR.

El principal beneficio de los modelos biomecánicos es su capacidad para proporcionar una visión detallada del funcionamiento de la rodilla sin necesidad de realizar experimentación física directa. Esto facilita una comprensión profunda de la mecánica articular y la predicción de comportamientos en diferentes condiciones. Los modelos biomecánicos resultan especialmente valiosos en el diseño de prótesis, la rehabilitación y la investigación clínica, ya que ofrecen una base sólida para el desarrollo de soluciones y tratamientos personalizados. En definitiva, los modelos biomecánicos son esenciales para calcular el CIR y comprender el movimiento de la rodilla a través de representaciones matemáticas precisas [4].

2.2.7.2.2 Modelos Biomecatrónicos

Los algoritmos de optimización son técnicas avanzadas utilizadas para ajustar los parámetros de modelos matemáticos mediante el análisis de datos experimentales sobre el movimiento de la rodilla, con el objetivo de determinar el Centro Instantáneo de Rotación. Estos algoritmos juegan un papel crucial en la mejora de la precisión y adaptabilidad de los modelos biomecatrónicos al identificar las configuraciones más adecuadas que se alinean con las observaciones reales.

Entre los métodos de optimización más comunes se encuentran los algoritmos genéticos. Basados en los principios de la evolución natural, estos algoritmos generan una población inicial de posibles soluciones para el modelo y las evalúan en función de su capacidad para ajustarse a los datos experimentales. A través de procesos iterativos de selección, cruzamiento y mutación, se crean nuevas generaciones de soluciones que mejoran progresivamente en términos de precisión. Este proceso permite encontrar las mejores configuraciones para los parámetros del modelo, lo que resulta en una representación más exacta del comportamiento de la rodilla[1].

Además de los algoritmos genéticos, existen otros métodos evolutivos aplicables en este ámbito, como los algoritmos de optimización por enjambre de partículas y los algoritmos de optimización por colonias de hormigas. Cada uno de estos enfoques emplea diferentes estrategias para explorar el espacio de soluciones y ajustar los parámetros del modelo. La selección del algoritmo apropiado depende de la complejidad del problema específico y de las características del modelo y los datos disponibles [2].

El uso de algoritmos de optimización es fundamental para calibrar modelos biomecánicos con precisión, permitiendo que los parámetros se ajusten de acuerdo a los datos experimentales y proporcionando una estimación precisa del CIR. Esta técnica resulta esencial para mejorar la exactitud de los modelos matemáticos y para obtener una comprensión detallada del movimiento de la rodilla en diversas condiciones y rangos de movimiento. En resumen, los algoritmos de optimización, incluidos los genéticos y otros métodos evolutivos, son cruciales para el ajuste y la validación de modelos biomecánicos basados en datos experimentales.

2.2.7.3 Métodos Experimentales en Laboratorio

La investigación del movimiento humano, especialmente en articulaciones como la rodilla, se beneficia enormemente de los métodos experimentales no invasivos. Mediante el uso de tecnologías avanzadas, como sistemas de captura de movimiento y sensores inerciales, los investigadores pueden analizar de forma detallada y precisa los patrones de movimiento. Estos métodos permiten construir modelos biomecánicos que simulan el comportamiento de la articulación, facilitando la comprensión de su función y el diseño de soluciones personalizadas para mejorar el rendimiento y la rehabilitación.

2.2.7.3.1 Método de Puntos de Referencia

En un entorno controlado, el empleo de dispositivos de medición especializados facilita la identificación de puntos clave en la rodilla durante su movimiento. Estos dispositivos pueden incluir cámaras de alta definición, sensores de posición y sistemas de captura de movimiento, que permiten registrar datos detallados sobre la ubicación y el desplazamiento de la articulación en distintas fases del movimiento. La información recopilada abarca coordenadas espaciales y ángulos articulares, esenciales para realizar un análisis exhaustivo.

Con los datos obtenidos, se procede al cálculo del Centro Instantáneo de Rotación de la rodilla, utilizando técnicas de geometría y análisis cinemático. La geometría se utiliza para construir un modelo estructural de la rodilla, teniendo en cuenta la disposición tridimensional de huesos y articulaciones. Estos modelos geométricos permiten entender cómo se relacionan los puntos de medición con el movimiento global de la articulación [21].

El análisis cinemático, por su parte, se concentra en cómo varían las posiciones y orientaciones de los componentes de la rodilla durante el movimiento. Aplicando principios cinemáticos, es posible determinar cómo las posiciones relativas de los puntos clave cambian y se relacionan con la rotación de la articulación. Esta información es crucial para calcular el CIR, el punto alrededor del cual la rodilla rota durante los movimientos [21].

La combinación de datos precisos obtenidos mediante mediciones directas y el uso de técnicas matemáticas avanzadas permite una evaluación precisa del CIR. Este enfoque no solo proporciona una comprensión detallada de la mecánica de la rodilla, sino que también es fundamental para aplicaciones en biomecánica, diseño de prótesis y planificación quirúrgica. En conclusión, el uso de dispositivos de medición en un entorno controlado, junto con técnicas geométricas y cinemáticas, facilita el cálculo exacto del CIR, mejorando la calidad de los análisis y las soluciones clínicas asociadas.

2.2.7.3.2 Método de Reconstrucción de Movimiento

El análisis de los movimientos de la rodilla a través de grabaciones de video de alta velocidad permite identificar con precisión el Centro Instantáneo de Rotación. Este método consiste en capturar imágenes de alta resolución que detallan los movimientos

articulares con gran claridad. Las cámaras de alta velocidad proporcionan un registro continuo y nítido de la dinámica de la rodilla en diversas fases del movimiento, permitiendo una evaluación exhaustiva de la articulación.

Para realizar este análisis, es necesario utilizar software especializado que procese las imágenes obtenidas y calcule el CIR. Este software convierte los datos visuales en información cuantitativa sobre los movimientos de la rodilla. Mediante algoritmos sofisticados, el software examina las secuencias de video y extrae datos precisos sobre la posición y el desplazamiento de la articulación. Estos algoritmos pueden rastrear los puntos de referencia en las imágenes y determinar cómo varían sus posiciones a lo largo del tiempo [22].

Una vez que se procesan los datos, el software calcula el CIR al analizar cómo cambia la rotación de la rodilla durante los movimientos observados. El cálculo del CIR se basa en la evaluación de las trayectorias de los puntos de referencia en relación con el eje de rotación de la rodilla. Este enfoque proporciona una estimación precisa del CIR al observar el comportamiento de la articulación durante la flexión y extensión [22].

En conclusión, el uso de grabaciones de video de alta velocidad junto con software especializado ofrece una herramienta eficaz para analizar los movimientos de la rodilla y determinar el CIR. Este método permite un estudio detallado y preciso de la mecánica articular, lo cual es crucial para aplicaciones en biomecánica, diseño de prótesis y planificación quirúrgica.

2.2.7.4 Métodos Computacionales

La simulación por computadora es una técnica avanzada que emplea representaciones digitales para examinar el movimiento de la rodilla, utilizando modelos biomecánicos detallados y datos específicos. Este enfoque permite generar modelos virtuales de la articulación y sus movimientos, proporcionando una herramienta eficaz para estudiar y entender la dinámica de la rodilla.

Para realizar una simulación por computadora, se introducen datos precisos sobre parámetros biomecánicos en un software especializado. Estos datos incluyen las dimensiones óseas, la configuración de los ligamentos y los rangos de movimiento esperados. El software crea una simulación digital de la rodilla y su comportamiento en

diversas posiciones y condiciones, proporcionando una representación exacta del funcionamiento articular [24].

El cálculo del Centro Instantáneo de Rotación se basa en el análisis de estas simulaciones digitales. Al observar cómo se comporta la rodilla en diferentes posiciones, el software determina el punto alrededor del cual la articulación rota, utilizando los datos modelados. Este proceso permite obtener una estimación precisa del CIR al simular la articulación en distintas fases del movimiento, ofreciendo una visión detallada de su comportamiento en diversas situaciones [24].

Una de las principales ventajas de la simulación por computadora es su capacidad para modelar y analizar el comportamiento de la rodilla sin necesidad de realizar pruebas físicas directas. Esto facilita un análisis profundo y la posibilidad de ajustar los modelos biomecánicos para mejorar su precisión. La simulación es particularmente valiosa en el diseño de prótesis, la planificación quirúrgica y la investigación biomecánica, ya que permite evaluar y optimizar soluciones y tratamientos de manera efectiva. En resumen, esta técnica ofrece una herramienta precisa para calcular el CIR y comprender el movimiento de la rodilla a través de simulaciones digitales detalladas.

2.2.7.5 Métodos de Evaluación Clínica

Las pruebas funcionales son una metodología que se utiliza para evaluar el movimiento de la rodilla mediante la observación y medición de sus movimientos durante actividades específicas. Este enfoque consiste en registrar cómo la articulación se comporta en situaciones reales o simuladas que imitan actividades cotidianas, tales como caminar, correr o subir escaleras. A través de estas pruebas, se recopilan datos valiosos sobre el rango de movimiento y el comportamiento de la rodilla en diferentes contextos, lo cual es fundamental para estimar el Centro Instantáneo de Rotación.

Durante las pruebas funcionales, se utilizan diversos equipos y técnicas para medir el movimiento de la rodilla. Estos pueden incluir dispositivos de medición como goniometros, cámaras de alta velocidad, o sistemas de captura de movimiento que registran los ángulos y posiciones articulares. La información obtenida de estas mediciones es luego analizada para estimar el CIR, que es el punto alrededor del cual la rodilla rota durante las actividades observadas [21].

Aunque las pruebas funcionales proporcionan información práctica y relevante sobre el funcionamiento de la rodilla en situaciones reales, pueden ser menos precisas en comparación con métodos directos o computacionales. La precisión de estas evaluaciones depende en gran medida de la calidad de los equipos utilizados y de la forma en que se llevan a cabo las pruebas. Además, la variabilidad en el comportamiento de la rodilla durante diferentes actividades puede influir en la estimación del CIR [21].

A pesar de sus limitaciones, las pruebas funcionales son valiosas porque ofrecen una perspectiva práctica del funcionamiento de la rodilla. Son especialmente útiles en contextos clínicos y rehabilitadores para evaluar la funcionalidad de la articulación y adaptar los tratamientos de manera más efectiva. En resumen, aunque menos precisas que otros métodos, las pruebas funcionales proporcionan datos importantes para la estimación del CIR y la comprensión del movimiento de la rodilla en condiciones prácticas.

Cada uno de estos métodos tiene sus ventajas y desventajas en términos de precisión, complejidad y recursos necesarios. La elección del método más adecuado dependerá del contexto específico y de los requisitos del estudio o aplicación en cuestión.

2.2.8 Selección del posible método de recolección de puntos

Para este estudio, se ha seleccionado el método de marcadores como la técnica más adecuada para la captura de datos cinemáticos de la rodilla. Esta elección se fundamenta en la precisión y versatilidad de este método, que permite identificar con alta exactitud el movimiento de puntos específicos de la articulación y calcular el Centro Instantáneo de Rotación. A continuación, se detallan las razones por las que se ha optado por este enfoque:

El método de marcadores ofrece una alta precisión en la captura de datos cinemáticos, gracias a la posibilidad de utilizar una amplia variedad de sistemas de medición, desde cámaras infrarrojas de alta velocidad hasta sensores electromagnéticos. La información detallada obtenida sobre la posición de los marcadores permite modelar de manera precisa el movimiento articular y calcular el Centro Instantáneo de Rotación, adaptándose a diferentes requerimientos experimentales

En este estudio, se ha seleccionado el método de marcadores por su capacidad de ofrecer un equilibrio óptimo entre precisión y versatilidad en la captura de datos cinemáticos. La información detallada obtenida a partir de estos marcadores será

fundamental para calcular el Centro Instantáneo de Rotación con alta precisión y optimizar el diseño del mecanismo de cuatro barras, garantizando así resultados confiables y aplicables a escenarios reales.

2.2.9 Algoritmos Evolutivos

Los algoritmos genéticos son técnicas computacionales que simulan los procesos evolutivos de la naturaleza para resolver problemas de optimización. Inspirados en la teoría de la evolución de Darwin, estos algoritmos manipulan una población de posibles soluciones, seleccionando y combinando las mejores para generar nuevas soluciones. A través de mecanismos como la selección natural, el cruce y la mutación, los algoritmos genéticos buscan encontrar soluciones óptimas o cercanas al óptimo para una amplia variedad de problemas [9].

2.2.9.1 Algoritmos genéticos.

Los algoritmos genéticos, inspirados en los procesos evolutivos de la naturaleza, son herramientas computacionales diseñadas para resolver problemas de optimización complejos. Al simular mecanismos como la selección natural, el cruce y la mutación, estos algoritmos exploran de manera eficiente un amplio espacio de soluciones. Cada posible solución es representada como un individuo dentro de una población, y su calidad se evalúa mediante una función de aptitud. A través de sucesivas generaciones, la población evoluciona hacia soluciones óptimas o cercanas al óptimo [9].

2.2.9.1.1 Estructura del Algoritmo genético.

En los algoritmos genéticos, la representación de una solución, conocida como cromosoma, es altamente adaptable y depende del problema específico. Esta representación determina cómo se codifica cada posible respuesta, permitiendo así que el algoritmo explore un amplio espacio de soluciones. La elección de la representación adecuada es crucial para el éxito de la búsqueda [24].

Representación de la solución

La representación cromosómica, núcleo de los algoritmos genéticos, es una estructura de datos que codifica de manera flexible las posibles soluciones a un problema. Esta representación, adaptable a la naturaleza de cada problema, influye directamente en la eficiencia y el éxito del proceso evolutivo. La elección de la representación adecuada

es un paso crucial en el diseño de un algoritmo genético, a continuación, se explica las formas de representación de los cromosomas.

Cadenas binarias: La representación binaria es una de las formas más comunes de codificar soluciones en algoritmos genéticos. En este esquema, cada posible solución se representa como una cadena de bits, donde cada bit puede tomar el valor de 0 o 1. Esta simplicidad permite una implementación eficiente de los operadores genéticos y facilita la adaptación a una amplia variedad de problemas. Pero puede ser menos útil para casos que no se prestan fácilmente a una representación binaria [24].

Vectores de números reales: Para problemas que demandan una mayor precisión o donde las variables varían de forma continua, la representación en punto flotante resulta ideal. En este enfoque, cada cromosoma es un vector de números reales, donde cada componente representa una variable del problema. Esta representación ofrece una mayor granularidad y flexibilidad, permitiendo explorar un espacio de búsqueda más amplio. Este caso es más adecuado para problemas con dominios continuos o múltiples variables interrelacionadas [24].

Inicialización de la población.

El algoritmo genético inicia su búsqueda de soluciones óptimas a partir de una población inicial, un conjunto aleatorio de individuos que representan posibles respuestas al problema. La diversidad y calidad de esta población inicial son fundamentales, ya que influyen directamente en la capacidad del algoritmo para explorar el espacio de soluciones y encontrar óptimos globales. Una buena inicialización asegura un punto de partida sólido para el proceso evolutivo [9]. A continuación, se presentan factores importantes a considerar para la creación de una población:

1) **Generación de la población inicial.**

   La creación de la población inicial es un paso fundamental en los algoritmos genéticos. Aunque la aleatoridad completa es un enfoque común, la estrategia óptima puede variar según la complejidad del problema. Algunas técnicas populares incluyen:

   Generación aleatoria: Cada individuo se construye asignando valores aleatorios a sus genes, garantizando una amplia exploración inicial.

Inicialización heurística: Se incorporan soluciones parciales o completas obtenidas mediante heurísticas o conocimiento previo del problema, acelerando la búsqueda en regiones prometedoras.

Combinación híbrida: Se combinan las dos estrategias anteriores, balanceando la exploración aleatoria con la explotación de información previa.

La elección de la estrategia de inicialización depende de factores como la complejidad del problema, la disponibilidad de información previa y los objetivos de la búsqueda [9].

2) **Tamaño de la población.**

La elección del tamaño de la población es un factor crucial en los algoritmos genéticos. Un tamaño adecuado debe encontrar un equilibrio entre la diversidad genética y el costo computacional. Una población más grande permite explorar un mayor rango de soluciones, reduciendo el riesgo de quedar atrapados en óptimos locales. Sin embargo, una población excesivamente grande puede ralentizar significativamente el proceso. Como regla general, para problemas sencillos, poblaciones entre 20 y 50 individuos suelen ser suficientes. Para problemas más complejos o de gran dimensión, se recomiendan poblaciones de 100 a 500 individuos. En casos donde las variables interactúan fuertemente, puede ser necesario aumentar aún más el tamaño de la población [9].

3) Factores de diversidad inicial

La diversidad de la población inicial es fundamental para el éxito de un algoritmo genético. Una población variada asegura que se explore un amplio rango de soluciones, reduciendo el riesgo de convergencia prematura hacia óptimos locales. Para lograr esta diversidad, se pueden emplear diversas estrategias:

Distribución uniforme: Los valores iniciales de los genes se asignan aleatoriamente dentro de sus rangos permitidos, garantizando una cobertura uniforme del espacio de búsqueda.

Muestreo estratificado: Esta técnica divide el espacio de búsqueda en regiones y asegura que cada región esté representada en la población inicial.

Verificación de factibilidad: Cuando el problema tiene restricciones, se verifica que cada individuo generado cumpla con dichas restricciones antes de ser incluido en la población [9].

4) Representación y factibilidad

Al generar la población inicial, es esencial considerar la representación utilizada para los individuos. La codificación debe permitir una representación adecuada del espacio de búsqueda y garantizar que los individuos generados sean válidos. En el caso de representaciones binarias, es necesario verificar que la longitud y el formato de los individuos sean correctos. Además, si el problema tiene restricciones, se deben implementar mecanismos para asegurar que los individuos iniciales cumplan con dichas restricciones. Si se genera un individuo inválido, se puede optar por descartarlo y generar uno nuevo, o bien, realizar modificaciones para hacerlo válido [9].

Función de aptitud (fitness function)

La función de aptitud es el compás que orienta la evolución de una población en un algoritmo genético. Esta función evalúa la calidad de cada solución candidata, determinando su adecuación para resolver el problema planteado. Al cuantificar el desempeño de cada individuo, la función de aptitud guía el proceso de selección, favoreciendo aquellos individuos que ofrecen mejores resultados y descartando los menos aptos.

La función de aptitud actúa como un evaluador, asignando a cada individuo un puntaje numérico que refleja su capacidad para resolver el problema planteado. Este puntaje, denominado valor de aptitud, determina la calidad de cada solución candidata. Dependiendo del problema, un valor de aptitud alto puede indicar una mejor solución (maximización) o una peor solución (minimización). El objetivo principal de la función de aptitud es establecer un criterio objetivo para comparar soluciones y dirigir el proceso evolutivo hacia aquellas que ofrecen mejores resultados [11]

A continuación se da a conocer cómo se diseña la función de aptitud dado que debe garantizar que:

Represente fielmente la calidad de cada solución candidata, reflejando de manera precisa su adecuación al problema.

Sea computacionalmente eficiente, ya que se evaluará repetidamente durante la ejecución del algoritmo.

Distinga claramente entre soluciones de diferentes calidades, evitando agrupar soluciones muy distintas en valores de aptitud similares.

Se adapte a diferentes escalas de calidad, permitiendo comparar soluciones que pueden variar significativamente en su desempeño.

Como siguiente punto para la función de aptitud tenemos la normalización de los valores de aptitud que es una práctica común en algoritmos genéticos, especialmente cuando se enfrentan a problemas de escala o valores negativos. Estas técnicas ayudan a evitar situaciones como:

Estancamiento: La falta de distinción entre individuos con valores de aptitud similares puede llevar a que la población converja prematuramente hacia soluciones subóptimas.

Dominancia prematura: Si un individuo es significativamente mejor que los demás, puede monopolizar la reproducción, reduciendo la diversidad de la población.

Como la normalización es un proceso que busca transformar los valores de aptitud a una escala común, facilitando la comparación entre individuos. Algunas técnicas de normalización comunes son:

Normalización min-max: Mapea los valores de aptitud al rango [0, 1] utilizando los valores mínimo y máximo de la población.

Escalado exponencial: Amplía las diferencias entre valores de aptitud utilizando una función exponencial.

Desplazamiento: Suma un valor constante a todos los valores de aptitud para asegurar que sean positivos

Cuando un problema de optimización incluye limitaciones o condiciones específicas, la función de aptitud debe adaptarse para manejar soluciones que no cumplen con estas restricciones. Algunas estrategias comunes son:

Sanciones: Asignar un valor de aptitud menor a las soluciones que infringen las reglas, penalizando así su calidad. Por ejemplo, en un problema de programación de tareas, una solución que genere conflictos en los horarios podría recibir una penalización.

Descarte: Eliminar del proceso evolutivo aquellas soluciones que no son válidas, evitando que influyan en las generaciones futuras.

Reparación: Modificar las soluciones inválidas para que cumplan con las restricciones antes de evaluar su aptitud. Esta estrategia implica realizar ajustes en la solución para hacerla factible [11].

Operadores genéticos

Los operadores genéticos son las herramientas que impulsan la evolución de una población en un algoritmo genético. Estos mecanismos permiten que los individuos interactúen y se modifiquen a lo largo de las generaciones, generando diversidad y buscando soluciones cada vez mejores. Los tres operadores fundamentales selección, cruce y mutación trabajan en conjunto para equilibrar la explotación de soluciones prometedoras y la exploración de nuevas regiones del espacio de búsqueda [25].

1) Selección

La selección es el mecanismo por el cual se eligen los individuos de una población que transmitirán sus características a la siguiente generación. Este proceso se basa en el principio de la supervivencia del más apto, aunque con un componente aleatorio para preservar la diversidad genética.

Métodos de selección comunes:

Selección proporcional: Los individuos con mayor aptitud tienen más probabilidades de ser seleccionados, similar a una ruleta donde el tamaño de cada sector representa la aptitud.

Selección por torneo: Se realizan pequeñas competiciones entre individuos seleccionados aleatoriamente, y el ganador pasa a la siguiente generación.

Selección de élite: Los mejores individuos se copian directamente a la siguiente generación para asegurar que no se pierdan las soluciones de alta calidad.

Selección por rango: Se asigna una probabilidad de selección basada en la posición relativa de un individuo dentro de la población, en lugar de su valor absoluto.

2) Cruce (Crossover)

El cruce es un operador genético que combina las características de dos o más individuos para crear nuevos descendientes. Este proceso es fundamental para explorar

nuevas regiones del espacio de búsqueda y aprovechar las mejores cualidades de los padres [24].

Tipos de cruce comunes:

Cruce de un punto: Se selecciona un punto de corte en los cromosomas de los padres y se intercambian los segmentos a partir de ese punto.

Cruce de dos puntos: Se seleccionan dos puntos de corte y se intercambia el material genético entre ellos.

Cruce uniforme: Cada gen se intercambia con una probabilidad fija, generando una mezcla más aleatoria.

Cruce específico: Se diseñan operadores de cruce adaptados a la estructura del problema, como en problemas de secuenciamiento o rutas.

3) Mutaciones

La mutación es un operador genético que introduce cambios aleatorios en los individuos de una población. Este mecanismo es esencial para mantener la diversidad genética y explorar nuevas regiones del espacio de búsqueda, evitando así que el algoritmo se estanque en soluciones subóptimas [11].

Tipos de mutación:

Inversión de bits: Se invierte el valor de uno o más bits en una representación binaria.

Intercambio de elementos: Se intercambian dos elementos en una representación de secuencia.

Perturbación numérica: Se añade un pequeño valor aleatorio a un valor numérico.

Mutación específica: Se diseñan mutaciones adaptadas a las características del problema

Reemplazo de la población

La sustitución generacional es un paso fundamental en los algoritmos genéticos que determina cómo se compone la siguiente población. Este proceso implica seleccionar los individuos que pasarán a la siguiente generación, combinando los progenitores con

sus descendientes. La estrategia de sustitución influye directamente en la diversidad genética de la población y en la velocidad con la que se encuentran soluciones óptimas. Existen dos enfoques principales para la sustitución generacional: la sustitución completa, donde toda la población es reemplazada, y la sustitución parcial, donde solo una parte de la población es sustituida.

En el enfoque de sustitución completa, la población entera es reemplazada por una nueva generación creada a partir de los individuos seleccionados y modificados mediante los operadores genéticos, este proceso tiene las siguientes ventajas [11].

Renovación total: Garantiza una exploración exhaustiva del espacio de búsqueda al reemplazar completamente la población.

Implementación sencilla: No requiere mecanismos complejos de comparación o combinación entre generaciones.

Pero puede presentar las siguientes desventajas.

Pérdida potencial de soluciones: Existe el riesgo de perder soluciones de alta calidad si no se implementan mecanismos adicionales para preservar los mejores individuos.

Homogeneización: Puede llevar a una disminución de la diversidad si los operadores genéticos no introducen suficiente variabilidad.

En el reemplazo parcial, solo una fracción de la población es sustituida por nuevos individuos en cada generación. Este enfoque busca equilibrar la explotación de las mejores soluciones encontradas (conservando los individuos más aptos) y la exploración de nuevas regiones del espacio de búsqueda (introduciendo nuevos descendientes).

Métodos comunes de reemplazo parcial:

Reemplazo generacional con élite: Se conservan los mejores individuos de la generación anterior y se completa la población con nuevos descendientes.

Reemplazo por nichos: Se priorizan individuos que aumenten la diversidad de la población, evitando la convergencia prematura.

Reemplazo basado en aptitud: Se seleccionan los mejores individuos de la población combinada (padres y descendientes).

Reemplazo estacionario: Se reemplaza una pequeña proporción de la población en cada generación, introduciendo cambios graduales.

Al seleccionar los individuos que pasarán a la siguiente generación, es fundamental considerar varios factores para garantizar un buen desempeño del algoritmo genético.

Preservación de los mejores: Es esencial conservar las soluciones de alta calidad para evitar perder los avances logrados.

Mantenimiento de la diversidad: Es importante mantener una diversidad genética suficiente para explorar diferentes regiones del espacio de búsqueda y evitar el estancamiento en soluciones subóptimas.

Velocidad de evolución: La estrategia de reemplazo influye directamente en la velocidad a la que el algoritmo converge hacia una solución.

Tamaño poblacional: El tamaño de la población afecta la elección de la estrategia de reemplazo, ya que poblaciones más pequeñas requieren mayor cuidado para preservar la diversidad.

CAPÍTULO III: MODELO DE LA INVESTIGACIÓN

3.1 Marco Metodológico

El presente estudio se enmarca en la metodología **cuantitativa**, específicamente en el enfoque de investigación **experimental**. Este enfoque se caracteriza por la manipulación deliberada de una variable independiente (en este caso, el algoritmo evolutivo) para observar su efecto sobre una variable dependiente específica (el diseño de mecanismos de 4 barras). Mediante la intervención controlada de las variables, se busca analizar su influencia y su relación con el objeto de estudio.

La investigación experimental se considera adecuada para este estudio debido a que permite:

Establecer relaciones de causa-efecto: Se podrá determinar si el algoritmo evolutivo desarrollado tiene un impacto significativo en la eficiencia del diseño de mecanismos de 4 barras.

Controlar variables: Se podrán controlar las variables que podrían influir en el diseño de los mecanismos, como las dimensiones de los eslabones, las posiciones de las articulaciones y las cargas aplicadas.

Medir variables de manera precisa: Se podrán medir variables cuantitativas relacionadas con el rendimiento de los mecanismos, como la velocidad, la aceleración y la fuerza.

3.2 Diseño de la investigación

La investigación se desarrollará siguiendo un diseño **factorial 2x2**, con dos factores:

Factor 1: Algoritmo evolutivo

Nivel 1: Algoritmo evolutivo genético

Nivel 2: Algoritmo evolutivo con basamento matemático

Factor 2: Tipo de mecanismo de 4 barras

Nivel 1: Mecanismo de cuatro barras simple

Nivel 2: Mecanismo de cuatro barras con restricciones

La investigación se llevará a cabo en las siguientes etapas:

Etapa 1: Revisión de literatura

Se realizará una revisión exhaustiva de la literatura existente sobre el diseño de mecanismos de 4 barras y los algoritmos evolutivos. Esto permitirá identificar el estado del arte en el tema y establecer la base teórica para el desarrollo del programa computacional.

Etapa 2: Desarrollo del programa computacional

Se desarrollará un programa computacional para el diseño de mecanismos de 4 barras basado en un algoritmo evolutivo. El programa deberá incluir una interfaz humano-máquina que permita al usuario ingresar los datos del problema y visualizar los resultados.

Etapa 3: Validación del programa computacional

Se validará el programa computacional mediante la comparación de sus resultados con los resultados de software comercial para el diseño de mecanismos de 4 barras.

Etapa 4: Aplicación del programa computacional

Se aplicará el programa computacional al diseño de una variedad de mecanismos de 4 barras, incluyendo mecanismos simples y mecanismos con restricción de movimiento. Se analizarán los resultados obtenidos y se compararán con los resultados obtenidos con métodos tradicionales de diseño.

Etapa 5: Análisis y discusión de resultados

Se analizarán los resultados obtenidos en las etapas anteriores y se discutirán las implicaciones de estos resultados para el diseño de mecanismos de 4 barras. Se formularán conclusiones y recomendaciones para futuras investigaciones.

3.3 Consideraciones adicionales

El programa computacional deberá ser desarrollado utilizando software libre para que pueda ser utilizado por la comunidad académica y la industria.

La interfaz humano-máquina deberá ser intuitiva y fácil de usar.

El programa computacional deberá ser eficiente en términos de tiempo de procesamiento.

Los resultados de la investigación deberán ser publicados en revistas científicas indexadas.

3.4 Rol de Python

El uso del lenguaje de programación **Python** será sido fundamental en el desarrollo de este trabajo. Python se caracteriza por su versatilidad, facilidad de uso y amplia comunidad de usuarios, lo que lo convierte en una herramienta ideal para la implementación de algoritmos genéticos y la creación de interfaces de usuario intuitivas.

El uso de Python garantiza que el programa desarrollado sea **accesible y práctico** para los usuarios, lo que contribuirá significativamente a su adopción y aplicación en el diseño de mecanismos de 4 barras.

En resumen, este estudio se basa en una metodología rigurosa y un diseño experimental sólido que permitirá comprender mejor la relación entre los algoritmos evolutivos y el diseño de mecanismos de 4 barras. La utilización de Python como herramienta de desarrollo garantizará la accesibilidad y practicidad del programa computacional, lo que contribuirá a su aplicación en el ámbito de la ingeniería y el diseño mecánico

3.5 Ética

La investigación se desarrollará de acuerdo con los principios éticos de la investigación científica. Se obtendrá el consentimiento informado de los participantes en la investigación y se protegerá la confidencialidad de sus datos.

3.6 Especificaciones del Sistema y Descripción de la Solución Propuesta

Se desarrollará una herramienta computacional basada en algoritmos evolutivos para el diseño personalizado de mecanismos de cuatro barras. Esta herramienta tiene como objetivo principal calcular el Centro Instantáneo de Rotación y optimizar el movimiento articular, con especial énfasis en aplicaciones biomecánicas como prótesis de rodilla policéntricas. La motivación detrás de este trabajo es la necesidad de soluciones personalizadas que se adapten a las características individuales de cada paciente, superando las limitaciones de los diseños genéricos disponibles en el mercado.

3.7. Especificaciones del Sistema

Para garantizar el éxito de la implementación y el uso del software desarrollado, se establecieron requisitos técnicos detallados tanto a nivel de hardware como de software. Estos requisitos buscan asegurar que la herramienta sea accesible, fácil de utilizar y eficiente, incluso en equipos con recursos computacionales limitados. El objetivo es proporcionar una solución práctica para el diseño de mecanismos de cuatro barras mediante algoritmos evolutivos, que permita obtener resultados precisos en un tiempo razonable

3.7.1 Requisitos de Hardware y Software

Para ejecutar el programa, se requiere un ordenador con un mínimo de 4 GB de RAM. El software necesario incluye Python 3.x, una versión moderna del lenguaje de programación Python. Adicionalmente, se deben instalar varias librerías esenciales: NumPy para operaciones numéricas avanzadas, SciPy para algoritmos y funciones adicionales en matemáticas y ciencia, y DEAP (Distributed Evolutionary Algorithms in Python) para implementar y ejecutar algoritmos evolutivos. Este conjunto de herramientas proporciona un entorno robusto y eficiente para desarrollar y ejecutar algoritmos genéticos

3.7.2 Entorno de Desarrollo

El desarrollo del programa se llevará a cabo en un entorno IDE como PyCharm o Jupyter Notebook, que permite una programación eficiente y pruebas iterativas. PyCharm ofrece potentes herramientas de desarrollo, incluyendo depuración y gestión de proyectos, mientras que Jupyter Notebook es ideal para pruebas rápidas y visualización interactiva de datos. La alternativa usada para este desarrollo fue IDLE, el entorno de desarrollo integrado que viene con Python. IDLE es ligero y fácil de usar, adecuado para la escritura de scripts y proyectos, proporcionando características básicas como resaltado de sintaxis y un depurador sencillo, facilitando el desarrollo ágil.

3.7.3 Interfaz Usuario-Máquina

Se diseñará una GUI (Graphical User Interface) intuitiva utilizando Tkinter o PyQt, facilitando la interacción del usuario con el programa. La interfaz permite la entrada de parámetros, visualización de resultados y ajustes en tiempo real. Tkinter es sencillo y se integra bien con Python, adecuado para interfaces básicas. PyQt, por otro lado, ofrece más flexibilidad y opciones avanzadas para crear interfaces complejas y estéticamente agradables. Ambas herramientas permiten desarrollar aplicaciones gráficas eficientes, mejorando la experiencia del usuario al proporcionar un entorno interactivo y fácil de usar para el diseño y optimización de mecanismos de 4 barras.

CAPÍTULO IV: RESULTADOS Y DISCUSIÓN

En este capítulo, se presenta un análisis detallado de los resultados obtenidos con el programa computacional modificado en PYTHON para el diseño de mecanismos de la articulación de rodilla, utilizando la curva del CIR como referencia. Se evalúan los resultados en función de los objetivos definidos, que incluyen la selección del lenguaje de programación (software libre), el desarrollo del software y su validación frente a curvas de movimiento CIR de casos de pacientes reales. La evaluación se realiza con un enfoque en la precisión, fiabilidad y consistencia del programa en comparación con los valores de referencia establecidos. Además, se examinan las limitaciones identificadas durante el proceso de desarrollo y se proponen posibles mejoras para optimizar el rendimiento del software. Este análisis exhaustivo proporciona una visión crítica del desempeño del programa, permitiendo una evaluación completa de su capacidad para diseñar mecanismos de rodilla. Las conclusiones extraídas del análisis ofrecen una comprensión clara de la eficacia del software y señalan áreas donde se puede mejorar, contribuyendo así a un mejoramiento continuo en el diseño de mecanismos de rodilla y en la aplicación de la curva del CIR en futuros desarrollos.

4.1 Descripción de la Solución Propuesta

El software se desarrolló con el objetivo de ofrecer una herramienta flexible y precisa. Permite a los usuarios introducir datos específicos para cada caso, como ángulos de movimiento y coordenadas del CIR. Además, incorpora mecanismos de normalización que garantizan la consistencia de los resultados, independientemente de las características de los dispositivos de medición empleados. Esto asegura la aplicabilidad de la herramienta en una amplia variedad de situaciones clínicas y de diseño

4.1.1 Funcionamiento del Algoritmo Evolutivo

El algoritmo evolutivo descrito en el código se utilizará para optimizar los parámetros de un mecanismo mediante un enfoque de selección natural. A continuación, se explica cómo será su funcionamiento paso a paso:

Paso 1

Lectura de Datos y Ajuste de Coordenadas:

Se leen los datos de coordenadas desde un archivo Excel.

Las coordenadas originales se trasladan a un nuevo rango definido para x e y.

Paso 2

Configuración de Parámetros Iniciales:

Se establecen valores iniciales para el ángulo máximo de flexión de la pierna y el tamaño de la población. El usuario podrá introducir este valor basándose en la medición realizada al paciente. Se recomienda probar el algoritmo con valores de ±3° y evaluar cuál proporciona el menor error para determinar el mecanismo más óptimo.

Cada individuo se representa mediante 6 cromosomas, donde cada cromosoma corresponde a un resultado a obtener. Estos cromosomas incluyen las medidas de los eslabones (a, b, c y d), el ángulo inicial y el ángulo final. Se genera una población inicial de 1000 individuos, un tamaño que se considera adecuado para equilibrar la precisión de los resultados y el rendimiento computacional. En caso de contar con mayores recursos, este número podría incrementarse en 1000 unidades. Los valores de los cromosomas se generan de forma aleatoria dentro de los rangos máximos y mínimos establecidos: -20 mm a 45 mm en el eje X y 25 mm a 200 mm en el eje Y. El programa se encarga de ajustar los valores de los puntos ingresados a estos rangos para asegurar el correcto funcionamiento del algoritmo evolutivo.

Paso 3

Evolución Genética:

El número de generaciones para la optimización se establece en 200, siguiendo la recomendación de Coello [26]. Este valor permite que el algoritmo evolucione durante un tiempo suficiente para explorar el espacio de soluciones y encontrar una solución óptima. Un mayor número de generaciones puede mejorar la precisión de las soluciones al refinar la adaptación de los cromosomas y, en caso de que el algoritmo no haya convergido, puede ayudar a evitar quedar atrapado en un óptimo local. En cada generación, se calcula el error de cada individuo comparando la curva deseada con la generada.

Paso 4

La población se ordena en función del error, desde los errores más grandes hasta los más pequeños. Luego, se seleccionan los individuos con los errores más bajos para

formar el conjunto élite, que serán los padres de la siguiente generación. Este proceso se repite iterativamente, seleccionando en cada generación los individuos que proporcionan el menor error para cada cromosoma.

Paso 5

Se generan 700 nuevos individuos mediante operaciones de mutación y cruce. Para la operación de cruce, se seleccionan aleatoriamente 6 individuos de la población actual. Cada nuevo individuo se forma combinando partes de los 6 individuos seleccionados: el primer valor del primer individuo, el segundo valor del segundo individuo, y así sucesivamente hasta que se toman los 6 valores de un individuo diferente. Este proceso genera una nueva población de individuos.

Paso 6

Actualización de la Población:

La nueva población generada por la élite, mutación y cruce reemplaza a la población original.

Paso 7

Extracción y Evaluación de Parámetros Óptimos:

Los parámetros del mejor individuo se extraen y se utilizan para generar la curva.

Se compara la curva generada con la curva deseada, calculando el error relativo absoluto para cada punto.

Paso 8

Visualización:

Se grafican las curvas deseada y generada, mostrando el proceso de evolución y optimización.

Paso 9

Cálculo del Error Total:

Se calcula el error euclidiano de la curva generada en comparación con la curva deseada.

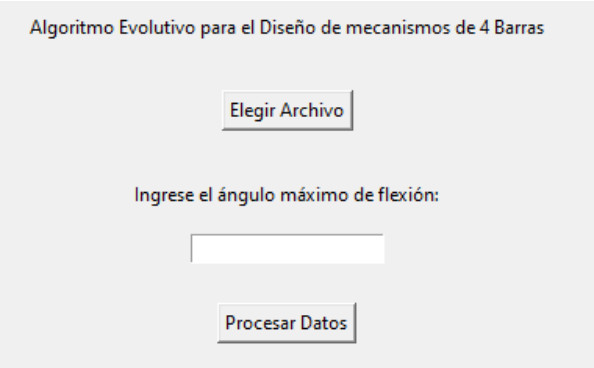
Este algoritmo evolutivo permite ajustar los parámetros del mecanismo para minimizar el error entre la curva generada y la curva deseada, mejorando el diseño a lo largo de las generaciones mediante la selección, mutación y cruce de individuos en la población.

4.1.2 Optimización del Diseño

Se emplean técnicas de optimización para ajustar los parámetros del mecanismo, garantizando que cumpla con los requisitos específicos del usuario. El algoritmo evolutivo iterativamente facilita el diseño del mecanismo, seleccionando y refinando las mejores configuraciones a lo largo de las generaciones. Esto permite a los usuarios ver claramente las mejoras en cada etapa, proporcionando una comprensión intuitiva de cómo el algoritmo ajusta y optimiza el diseño para satisfacer las necesidades específicas del paciente.

Figura 7.

Interfaz interactiva para el uso del Algoritmo genético



En la Figura 7 se muestra la interfaz humano-máquina con la que los usuarios interactuarán. Se trata de un diseño sencillo y altamente intuitivo. Al presionar el botón "Elegir archivo", se abre una ventana en la que se debe seleccionar un archivo en formato .xlsx, el cual debe contener dos columnas con los valores x e y del CIR del paciente. El rendimiento del algoritmo mejora con una mayor cantidad de datos, aunque puede operar con un mínimo de 10 puntos y un máximo de 500 puntos.

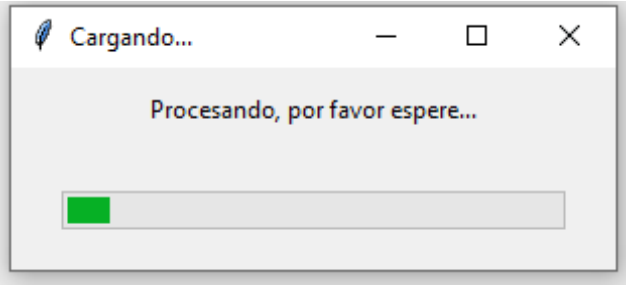
A continuación, se encuentra el recuadro para ingresar el ángulo máximo de flexión. Al igual que en el caso anterior, un ángulo mayor favorece el desempeño del algoritmo. Este puede operar con valores entre 2 y 180 grados, aunque para un funcionamiento óptimo se recomienda utilizar un ángulo superior a 100 grados. La

elección del ángulo dependerá de los métodos empleados para recopilar los puntos del CIR del paciente, así como del ángulo resultante de su movimiento.

Una vez ingresados todos los datos, se debe presionar el botón "Procesar datos". Inmediatamente, se abrirá una nueva ventana con una barra de progreso que indicará el avance del proceso como se indica en la Figura 8. Es importante esperar a que la barra se complete, ya que la ventana se cerrará automáticamente al finalizar. Posteriormente, aparecerá otra ventana confirmando que el proceso ha concluido con éxito. Se recomienda no realizar ninguna otra acción mientras la barra de progreso está en ejecución, ya que esto podría consumir demasiados recursos del sistema y provocar el cierre forzado del programa. Si ocurre algún error, el programa lo notificará en una ventana emergente. En caso de no recibir ninguna notificación de error, se debe cerrar la ventana de confirmación para proceder a visualizar los resultados y gráficas.

Figura 8.

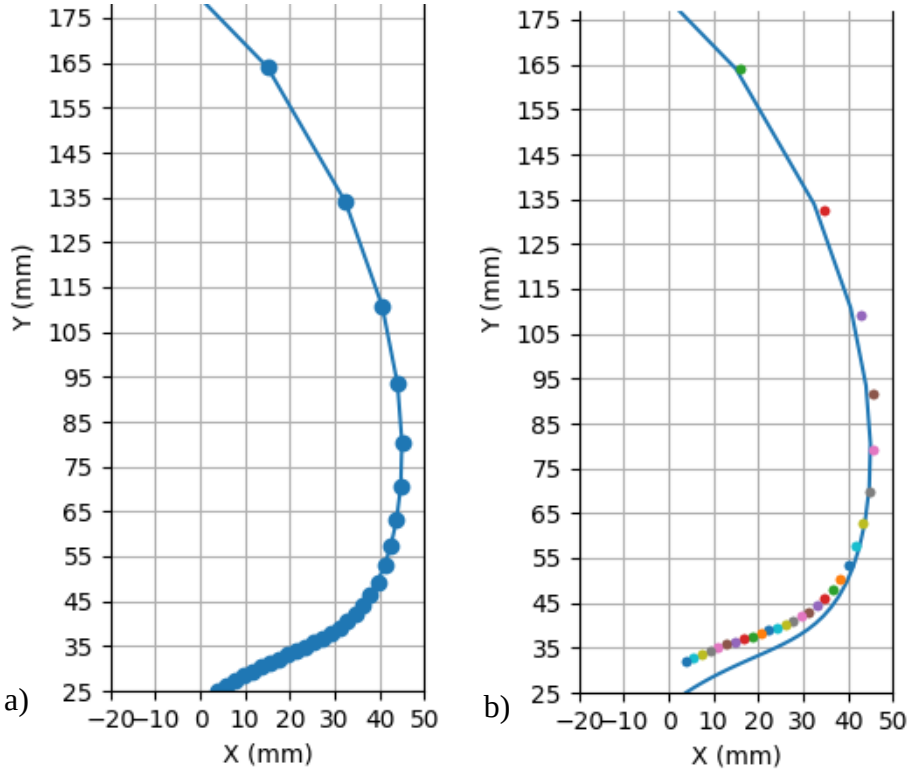
Ventana que indica el progreso



Además, la obtención de coordenadas para la curva ideal puede ubicarse en cualquier parte del plano cartesiano. Para normalizar estas posiciones, como se dijo anteriormente, se implementó un código inicial que desplaza las coordenadas **x** e **y** desde su ubicación original a una posición estándar. Este proceso de translación asegura que todas las coordenadas se ajusten a un punto de referencia común (población inicial), facilitando su uso en el algoritmo evolutivo. Esta normalización es crucial para mantener la consistencia y precisión en los cálculos posteriores, permitiendo que el algoritmo trabaje con datos coherentes y optimice el diseño del mecanismo de manera efectiva y eficiente como podemos observar en la Figura 9.

Figura 9.

Gráficas impresas en la interfaz



4.2 Arquitectura del algoritmo

El código que se ha analizado presenta una arquitectura híbrida que integra de manera eficiente elementos característicos de las aplicaciones de escritorio tradicionales con funcionalidades propias de herramientas de cálculo científico y matemático. Esta combinación única permite al usuario interactuar con una interfaz gráfica intuitiva, similar a la de cualquier programa, al tiempo que ejecuta complejos cálculos numéricos y simulaciones. A continuación, se detallará la estructura general de esta aplicación, explorando cómo se han integrado estos componentes para lograr una solución versátil y eficiente.

4.2.1 Arquitectura General

La arquitectura se puede describir como modular monolítica, dado que todo el código está centralizado en un único archivo o script, aunque hay ciertos intentos de separación por funcionalidad.

4.2.2 Componentes y Módulos Principales

La aplicación está estructurada en capas bien definidas para facilitar su desarrollo y mantenimiento. La capa de presentación, desarrollada con tkinter, proporciona una interfaz gráfica intuitiva. La capa de negocio se encarga de las operaciones de alto nivel, como la gestión de datos y la interacción con el usuario. La capa de cálculo, que utiliza librerías como numpy y math, implementa los algoritmos matemáticos y de optimización. Finalmente, la capa de integración coordina el flujo de datos entre las diferentes capas y gestiona la ejecución concurrente de tareas mediante threading.

4.2.2.1. Capa de Presentación (Interfaz Gráfica de Usuario - GUI)

Se usa la biblioteca tkinter para construir la interfaz gráfica de usuario.

Incluye elementos como:

Ventanas principales y secundarias (root, Toplevel) para la interfaz y ventanas de progreso.

Widgets gráficos, como:

- Botones (Button) para activar procesos.
- Barras de progreso (ttk.Progressbar) para indicar estados.
- Etiquetas (Label) para mostrar texto y resultados.
- Contenedores (Frame) para organizar elementos.

Carga y visualización de imágenes utilizando PIL (Pillow).

Embeber gráficos generados con matplotlib en la interfaz usando FigureCanvasTkAgg.

4.2.2.2 Capa de Lógica de Negocio

Procesamiento de datos:

Lectura de datos desde un archivo Excel usando pandas.

Escalado de datos (transformaciones lineales de coordenadas x e y).

Escritura de los datos modificados en un nuevo archivo Excel.

Visualización:

Generación de gráficas con matplotlib para representar datos originales y modificados.

Comparación de datos generados y reales.

Algoritmo evolutivo:

Se implementa una técnica evolutiva para optimizar un conjunto de parámetros mediante simulación:

Inicialización de una población.

Mutaciones, cruces y evaluación de generaciones.

Selección de individuos con mejor desempeño.

4.2.2.3. Capa de Cálculo Científico

Utiliza bibliotecas como numpy y math para operaciones matemáticas y científicas:

Manipulación de matrices y vectores.

Cálculo de errores entre datos reales y simulados.

Evaluación de funciones de ajuste (E_CIR y CIR).

4.2.2.4. Capa de Integración y Control

Gestión de hilos:

Usa threading para realizar tareas simultáneas:

Mostrar una barra de progreso mientras se procesan datos.

Ejecutar cálculos en segundo plano sin bloquear la interfaz gráfica.

Manejo de excepciones:

Usa bloques try-except para manejar errores, como fallos al leer archivos o cálculos matemáticos.

4.2.3. Flujo General del Programa

Inicio de la Aplicación:

La ventana principal (root) se muestra con opciones para cargar archivos, iniciar procesos y visualizar resultados.

Carga de Archivo:

El usuario selecciona un archivo Excel mediante un cuadro de diálogo.

Procesamiento de Datos:

Los datos se escalan y transforman.

Los datos ajustados se escriben en un nuevo archivo Excel.

Ejecución del Algoritmo Evolutivo:

Se optimizan parámetros para ajustar datos simulados a los ideales.

Visualización:

Se muestran gráficas de los datos originales, transformados y simulados en la interfaz.

Progreso y Feedback:

La barra de progreso indica el estado de procesamiento.

Se muestran mensajes al usuario en caso de éxito o error.

4.2.4. Limitaciones y Observaciones

Monolítico: Todo el código está en un solo archivo, lo que puede dificultar el mantenimiento y la escalabilidad.

Acoplamiento Alto: Las funciones de GUI, procesamiento de datos y cálculos científicos están muy entrelazadas.

Falta de Abstracción: No hay separación clara entre capas lógicas, lo que dificulta la reutilización de componentes.

Uso de Hilos: Aunque se utiliza threading, no hay manejo avanzado de sincronización, lo que podría llevar a problemas en aplicaciones más complejas.

4.3 Seudocódigo de Python

```
INICIO
  IMPORTAR las bibliotecas necesarias: PIL, tkinter, numpy, pandas, matplotlib, math, threading, time.
  DEFINIR función `process_data`:
    TRATAR:
      OBTENER el ángulo desde la entrada de la interfaz.
      CONVERTIR el ángulo a radianes.
      LEER los datos desde un archivo Excel.
```

```
        PROCESAR los datos:
            SEPARAR los valores x e y.
            ESCALAR y DESPLAZAR las coordenadas x e y al nuevo rango.
        GUARDAR los nuevos datos en un archivo Excel.
        CREAR la primera gráfica con los datos ajustados.
        EMBEBER la gráfica en el lado izquierdo de la interfaz.
        INICIALIZAR población aleatoria para el algoritmo evolutivo.
        DEFINIR función de evaluación `E_CIR`.
        REALIZAR algoritmo evolutivo:
            PARA cada generación:
                EVALUAR calidad de cada individuo.
                ORDENAR la población.
                GENERAR nueva población con mutación y cruces.
        EXTRAER los mejores valores de parámetros.
        DEFINIR función `CIR` para calcular coordenadas.
        OBTENER las coordenadas ajustadas usando los parámetros obtenidos.
        CREAR la segunda gráfica comparando datos ideales y generados.
        EMBEBER la gráfica en el lado derecho de la interfaz.
        CALCULAR error total (ET).
        ACTUALIZAR etiquetas con los resultados.
        MOSTRAR mensaje de éxito.
    CAPTURAR excepción y MOSTRAR mensaje de error.
DEFINIR función `display_image`:
    TRATAR:
        CARGAR y REDIMENSIONAR una imagen.
        MOSTRAR la imagen en la interfaz.
    CAPTURAR excepción y MOSTRAR mensaje de error.
DEFINIR función `show_progress_bar`:
    CREAR ventana de progreso.
    MOSTRAR barra de progreso durante un tiempo fijo.
    CERRAR ventana al finalizar.
DEFINIR función `process_excel`:
    OBTENER la ruta del archivo Excel del usuario.
DEFINIR función `start_process`:
    INICIAR hilo para mostrar la barra de progreso.
    INICIAR hilo para ejecutar la tarea principal.
```

CREAR ventana principal de la aplicación.

CONFIGURAR elementos de la interfaz:

Entradas de texto.

Botones para seleccionar archivo, procesar datos y mostrar imagen.

Etiquetas para mostrar resultados.

EJECUTAR el bucle principal de la interfaz gráfica.

FIN

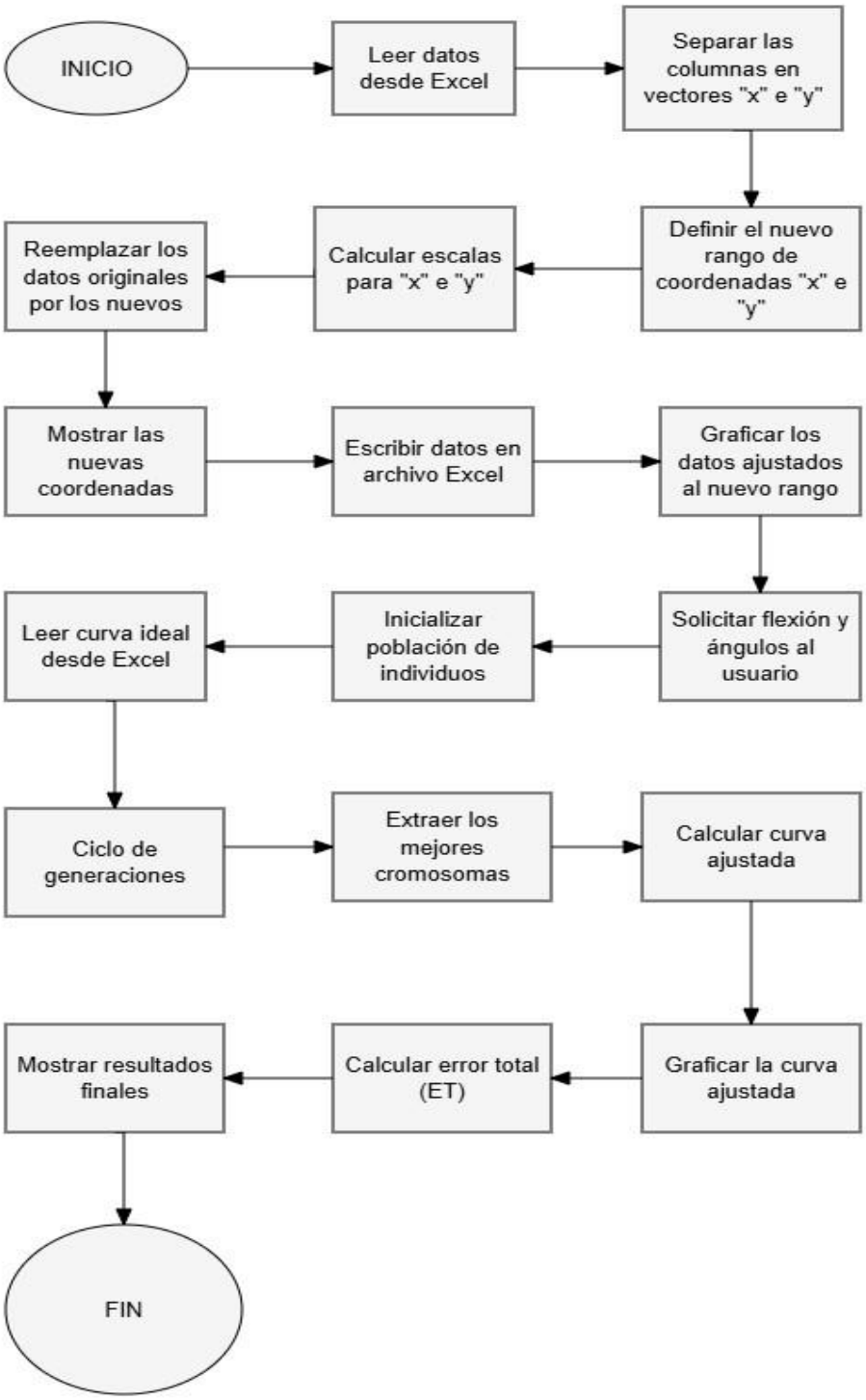
4.4 Desarrollo del programa computacional tomando en cuenta el algoritmo genético base desarrollado en Matlab.

Después de llevar a cabo una revisión detallada y un análisis exhaustivo del código original escrito en MATLAB, se ha procedido a reescribirlo en Python. Esta reescritura ha permitido transformar el código para que sea adaptable a diferentes tipos de curvas y conjuntos de datos. La nueva versión en Python no solo mantiene la funcionalidad del código original, sino que también introduce una mayor flexibilidad al permitir la entrada de datos mediante el teclado, como los ángulos máximos de flexión o los ángulos iniciales. Esta capacidad para ajustar parámetros en tiempo real mejora la versatilidad del programa y facilita su uso en una amplia gama de aplicaciones. La adaptación a Python ha optimizado la ejecución y ha asegurado que el programa pueda gestionar eficientemente una variedad de datos, haciendo que el proceso de diseño y análisis sea más accesible y dinámico para el usuario. Esta modernización no solo mantiene la precisión en el modelado de mecanismos, sino que también enriquece la experiencia del usuario al ofrecer una interfaz más interactiva y adaptable.

En el desarrollo del programa computacional, se consideró fundamental elaborar un diagrama de flujo, como se muestra en la Figura 10. Este diagrama detalla el procedimiento seguido por los algoritmos genéticos para alcanzar una solución óptima.

Figura 10.

Diagrama de Flujo del algoritmo evolutivo



Descripción detallada del diagrama de flujo:

1. **Inicio:**

    - Comienza el proceso.

2. **Procesamiento de Datos:**

- Leer Datos: Se leen los datos desde un archivo Excel.
- Separar Columnas: Los datos se dividen en vectores x_original y y_original.
- Definir Nuevos Rangos: Se establecen los nuevos rangos para x y y.
- Calcular Escala y Desplazamiento: Se ajustan los valores de x e y a los nuevos rangos.
- Reemplazar Datos: Los datos originales se reemplazan por los ajustados.
- Mostrar Nuevos Datos: Se despliegan los nuevos datos en la consola.
- Escribir Datos en Archivo: Se guardan los datos ajustados en un nuevo archivo Excel.
- Graficar Datos: Se grafica la curva ajustada.

3. **Algoritmo Genético:**

- Solicitar Entradas: Se piden al usuario los parámetros necesarios.
- Inicializar Población: Se crea la población inicial y se configuran los parámetros del algoritmo genético.
- Leer Curva Deseada: Se lee la curva ideal desde un archivo Excel.
- Bucle de Generaciones:
- Calcular el error para cada individuo.
- Ordenar y seleccionar la población.
- Crear nueva población mediante elitismo, mutación y cruce.
- Extraer y Calcular: Se extraen los mejores cromosomas y se ajusta la curva.
- Graficar y Calcular Error: Se grafica la curva ajustada y se calcula el error total.
- Mostrar Resultados: Se muestran los resultados finales.

4. **Fin:**

- Termina el proceso.

Este diagrama de flujo proporciona una visión clara del proceso que sigue el código desde la manipulación inicial de datos hasta la aplicación del algoritmo genético y la visualización de resultados.

El siguiente capítulo cierra la investigación y ofrece un resumen de los hallazgos y logros obtenidos en el desarrollo del software para diseñar el mecanismo de la articulación de la rodilla basado en la curva CIR. En este capítulo, se presentan las conclusiones obtenidas tras analizar los resultados y se destaca la exactitud y fiabilidad del programa en la determinación del CIR. Se proporciona un resumen de las aportaciones realizadas a la biomecánica de la rodilla y se examinan las implicaciones prácticas y teóricas de la investigación. Además, se incluyen recomendaciones para trabajos futuros, tales como la mejora de la implementación del software, la investigación de nuevas técnicas de optimización y la aplicación del programa en contextos clínicos o en el diseño de prótesis. Estas conclusiones y sugerencias ofrecen una visión integral y definitiva del trabajo realizado, completando así el ciclo de investigación y ofreciendo orientación para futuras investigaciones en este campo.

4.5 Estructuración del código en un software libre (Python)

En el desarrollo de este proyecto, la elección de un software libre como Python para la estructuración del código fue una decisión clave que permitió garantizar la accesibilidad, flexibilidad y escalabilidad del programa computacional. Python es reconocido por su simplicidad, versatilidad y una amplia comunidad de desarrollo, lo que lo convierte en una herramienta ideal para implementar algoritmos evolutivos y trabajar con datos biomecánicos complejos.

La estructuración del código en Python se enfocó en traducir y optimizar un algoritmo previamente desarrollado en MATLAB, adaptándolo a las capacidades y bibliotecas disponibles en este lenguaje de programación. Durante este proceso, se reorganizaron las funciones para que estuvieran contenidas dentro de un mismo archivo, mejorando la modularidad y la eficiencia del programa. Además, la integración de bibliotecas como NumPy, SciPy y DEAP facilitó la implementación de cálculos numéricos avanzados, optimización y generación de poblaciones evolutivas.

El enfoque en software libre no solo asegura que el programa sea accesible para una amplia comunidad de usuarios e investigadores, sino que también fomenta la colaboración y mejora continua. En este apartado, se detalla cómo se estructuró el código para aprovechar al máximo las capacidades de Python, garantizando un rendimiento eficiente y la adaptabilidad necesaria para aplicaciones biomecánicas personalizadas

4.5.1 Proceso de Conversión de MATLAB a Python

La conversión del código de MATLAB a Python implicó un proceso meticuloso de traducción línea por línea. MATLAB, conocido por su sintaxis especializada en matrices y su entorno interactivo, se basa en funciones que a menudo operan de manera autónoma. En contraste, Python ofrece una sintaxis más general y flexible. Este cambio presentó la oportunidad de consolidar funciones en un solo archivo de código, lo que facilita la gestión y la comprensión del flujo de trabajo. En MATLAB, las funciones se almacenan por separado, lo que requiere llamadas explícitas y, a menudo, interrumpe el flujo del código. Sin embargo, al trasladar el código a Python, se optó por integrar funciones dentro del mismo script principal, beneficiándose de una mayor cohesión y reducción de dependencias externas. Este enfoque no solo simplificó el desarrollo, sino que también mejoró la eficiencia del código. Además, Python permite la creación de módulos que encapsulan la funcionalidad específica, lo cual contribuye a un código más modular y reutilizable. Así, el proceso de conversión no solo implicó traducir comandos, sino también optimizar la estructura del código para aprovechar las características únicas de Python, como la capacidad de definir funciones locales y globales dentro de un solo archivo, facilitando la implementación y el mantenimiento del código.

4.5.2 Facilidad de Python trabajando con Algoritmos Evolutivos

Python se destaca por su capacidad para manejar algoritmos complejos de manera eficiente, lo que lo convierte en una excelente opción para la implementación de algoritmos evolutivos. A diferencia de MATLAB, que tiene una sintaxis centrada en matrices y una curva de aprendizaje específica, Python ofrece una sintaxis más intuitiva y legible. Esta facilidad de uso es particularmente valiosa cuando se trabaja con algoritmos evolutivos, que requieren la manipulación de grandes conjuntos de datos y la ejecución de operaciones repetitivas. La versatilidad de Python se refleja en su amplia gama de bibliotecas especializadas, como NumPy para operaciones matemáticas avanzadas y SciPy para funciones científicas adicionales. En particular, la biblioteca DEAP (Distributed Evolutionary Algorithms in Python) se destaca por su capacidad para implementar y ejecutar algoritmos evolutivos de manera efectiva. DEAP proporciona una estructura modular que facilita la creación de poblaciones, la evaluación de fitness y la aplicación de operadores genéticos como cruce y mutación. Esta capacidad de Python para integrarse con diversas bibliotecas y su enfoque en la simplicidad del código hacen

que la implementación de algoritmos evolutivos sea más accesible y menos propensa a errores como se muestra en la Figura 11.

Figura 11.

Ecuaciones de Freudstein colocadas en Python

```
# Función de evaluación E_CIR (debes definirla)
def E_CIR(cir_ideal, k, tr_max):
    a = k[0]
    b = k[1]
    c = k[2]
    d = k[3]
    t1 = k[4]
    t_as = k[5]

    X_cir_i = cir_ideal[:, 0]
    Y_cir_i = cir_ideal[:, 1]

    n = len(X_cir_i)

    t_r = np.linspace(0, tr_max, n)
    t3 = t_as + t_r

    A = (a**2 + b**2 - c**2 + d**2) / (2 * a * b) + (d / b) * np.cos(t1) - (d / a) * np.cos(t1 - t3) - np.cos(t3)
    B = 2 * (np.sin(t3) - (d / b) * np.sin(t1))
    C = (a**2 + b**2 - c**2 + d**2) / (2 * a * b) - (d / b) * np.cos(t1) - (d / a) * np.cos(t1 - t3) + np.cos(t3)

    D = (-a**2 + b**2 + c**2 + d**2) / (2 * b * c) - d * (np.cos(t1 - t3)) / c - d * np.cos(t1) / b + np.cos(t3)
    E = 2 * (d * np.sin(t1) / b - np.sin(t3))
    F = (-a**2 + b**2 + c**2 + d**2) / (2 * b * c) - d * (np.cos(t1 - t3)) / c + d * np.cos(t1) / b - np.cos(t3)

    t2 = 2 * np.arctan(0.5 * (-B + np.sqrt(B**2 - 4 * A * C)) / A)
    t4 = 2 * np.arctan(0.5 * (-E - np.sqrt(E**2 - 4 * D * F)) / D)

    Xoa = 0
    Yoa = 0
    Xob = d * np.cos(t1)
    Yob = d * np.sin(t1)
    Xa = a * np.cos(t2)
    Ya = a * np.sin(t2)
    Xb = Xob + c * np.cos(t4)
    Yb = Yob + c * np.sin(t4)

    Xb_cir = (Xob * np.tan(t4) - Yob) / (np.tan(t4) - np.tan(t2)) - d * np.cos(t1) - c * np.cos(t4)
    Yb_cir = (Xob * np.tan(t4) - Yob) * np.tan(t2) / (np.tan(t4) - np.tan(t2)) - d * np.sin(t1) - c * np.sin(t4)
```

4.5.3 Paso a Paso del Código Funcional en Python

El código funcional en Python para el diseño del mecanismo de rodilla basado en algoritmos evolutivos sigue un proceso sistemático que abarca desde la inicialización hasta la evaluación de la solución óptima. En primer lugar, se lee el conjunto de datos que define la curva deseada, la cual se puede importar desde un archivo CSV o Excel utilizando bibliotecas como pandas. A continuación, se genera una población inicial de soluciones aleatorias, en la que cada individuo representa una configuración posible del mecanismo de cuatro barras. La evaluación de fitness se realiza comparando la curva generada por el mecanismo con la curva deseada, utilizando métricas de error como el error euclideano. Posteriormente, se aplican los operadores genéticos: selección, cruce y mutación, para producir nuevas generaciones de soluciones. La selección se basa en el rendimiento de los individuos, el cruce combina características de los mejores individuos

y la mutación introduce variaciones aleatorias para explorar nuevas soluciones. Este ciclo iterativo continúa hasta que se alcanza un criterio de convergencia, como un número fijo de generaciones o un umbral de error satisfactorio. Finalmente, el código muestra las curvas generada y deseada para visualizar la calidad de la solución óptima encontrada.

4.5.4 Funciones Auxiliares y Aplicación de la Ecuación de Freudenstein

El código en Python incluye varias funciones auxiliares que simplifican y organizan el proceso de implementación del algoritmo evolutivo. Estas funciones abarcan desde la generación de la población inicial hasta la aplicación de operadores genéticos y la evaluación de fitness. Una de las funciones clave es la que implementa la ecuación de Freudenstein, utilizada para calcular las dimensiones de los eslabones en un mecanismo de cuatro barras. Esta ecuación se aplica en el contexto del problema para determinar cómo ajustar las longitudes de los eslabones para que la curva generada coincida con la curva deseada. La implementación de la ecuación de Freudenstein en Python se realiza utilizando bibliotecas matemáticas como NumPy para las operaciones algebraicas y SciPy para la optimización. El cálculo del error euleriano se utiliza para medir la precisión de la curva generada, comparando la diferencia entre los puntos de la curva deseada y la curva generada por el mecanismo. Este análisis de error proporciona una métrica clara de la calidad de la solución encontrada y permite ajustar el algoritmo para mejorar la precisión. Así, el uso de funciones auxiliares y la aplicación precisa de la ecuación de Freudenstein en Python garantizan un diseño eficiente y efectivo del mecanismo, con una evaluación robusta de los resultados obtenidos.

4.6 Optimización del código base

La optimización del código base fue un paso crucial para mejorar la eficiencia, flexibilidad y adaptabilidad del programa computacional diseñado. Este proceso implicó ajustar y perfeccionar las funcionalidades del algoritmo original para asegurar que pudiera manejar una variedad de conjuntos de datos y parámetros personalizados, garantizando al mismo tiempo la precisión de los cálculos.

Uno de los principales enfoques de la optimización fue implementar herramientas que permitieran la normalización de coordenadas y el ingreso manual de valores clave, como ángulos máximos de extensión o límites de movimiento. Estas mejoras no solo facilitaron la personalización del software para diferentes usuarios, sino que también

aseguraron que las curvas generadas conservaran su forma original, independientemente del dispositivo de medición o la fuente de datos.

Durante este proceso, se aprovecharon al máximo las capacidades del lenguaje de programación Python y sus bibliotecas especializadas, como NumPy y SciPy, para realizar cálculos numéricos avanzados de manera más eficiente. Asimismo, se introdujeron funciones auxiliares para simplificar tareas repetitivas y mejorar la modularidad del código, haciéndolo más fácil de mantener y expandir en el futuro.

En este apartado, se describen las estrategias y ajustes realizados para optimizar el código base, destacando cómo estas mejoras contribuyeron a la precisión y funcionalidad del programa en aplicaciones biomecánicas.

4.6.1 Desplazamiento de coordenadas hasta la población inicial

Durante el proceso de desarrollo del programa para el diseño del mecanismo de la articulación de la rodilla basado en la curva CIR, se identificó la necesidad de optimizar el código inicial para manejar una mayor variedad de conjuntos de datos. Originalmente, el código estaba diseñado para trabajar con un conjunto específico de coordenadas, lo cual limitaba su flexibilidad y aplicabilidad a diferentes contextos. Dado que las coordenadas **x** e **y** del CIR pueden variar dependiendo de la fuente de los datos, ya sea de diferentes personas o de distintos aparatos de medición, era crucial adaptar el algoritmo para que pudiera ajustarse a cualquier variación en los datos sin comprometer la integridad de la curva.

La optimización se centró en desarrollar un nuevo código en Python que permitiera trasladar las coordenadas de entrada. Este enfoque era fundamental para mantener la forma original de la curva mientras se ajustaba a la población inicial definida en el algoritmo evolutivo. La idea principal era desplazar las coordenadas sin alterar su forma, para que la curva pudiera alinearse adecuadamente con la población inicial del algoritmo, garantizando que el proceso de optimización se realizara de manera precisa y efectiva.

El nuevo código implementado realiza un desplazamiento de las coordenadas “X” e “Y” desde su ubicación original hacia una posición estándar, correspondiente a la población inicial. Esto se logra mediante técnicas de normalización que trasladan los puntos de datos a un punto de referencia común. Este desplazamiento asegura que el

algoritmo evolutivo pueda trabajar con una mayor cantidad de conjuntos de datos, independientemente de las variaciones en la medición o en la obtención de puntos, sin que la curva pierda su forma característica.

Esta optimización permite que el programa sea más flexible y adaptable, facilitando su uso en una gama más amplia de aplicaciones y contextos. Al mantener la forma original de la curva mientras se ajusta a la población inicial, se mejora la precisión y la eficacia del algoritmo evolutivo, asegurando que los resultados sean consistentes y fiables, sin importar las variaciones en los datos de entrada. En resumen, esta actualización en el código amplía significativamente la versatilidad del programa, permitiendo su aplicación en diferentes escenarios y con distintos conjuntos de datos.

4.6.2 Ingreso por teclado de variables

La optimización del código para permitir la entrada del ángulo máximo de extensión de la pierna por teclado ha sido una necesidad clave en el desarrollo del programa. Esta mejora es fundamental debido a que el conjunto de datos utilizado puede variar considerablemente en cuanto al rango de movimiento de la pierna. En muchos casos, los datos no abarcan un rango completo de 0 a 180 grados, sino que pueden variar desde ángulos mucho menores, como 3 grados, hasta ángulos más amplios. Por ejemplo, uno de los pacientes en los estudios mostró un rango de movimiento limitado, lo que requiere que el programa se adapte a estos valores específicos.

Implementar una funcionalidad que permita ingresar el ángulo máximo de extensión por teclado facilita una mayor flexibilidad en la adaptación a diferentes conjuntos de datos. Esto asegura que el programa pueda ajustarse a las variaciones individuales de los pacientes y mantener la precisión en el diseño del mecanismo.

Además, es crucial ajustar también la población en el algoritmo evolutivo para que los límites de los ángulos sean configurables de manera similar. Si el ángulo de extensión se reduce, el rango de movimiento del mecanismo se verá afectado, lo que subraya la importancia de permitir ajustes dinámicos en los parámetros del algoritmo.

Este enfoque de optimización asegura que el programa pueda manejar eficazmente diferentes rangos de datos y adaptarse a las necesidades específicas del paciente, mejorando así la precisión del diseño y la funcionalidad del mecanismo. La capacidad de ajustar los ángulos y la población de manera flexible permite al programa generar

soluciones más adecuadas y personalizadas, optimizando el rendimiento y la aplicabilidad del diseño en contextos clínicos y de investigación. En resumen, realizar estos cambios en el código es esencial para manejar la variabilidad en los datos y asegurar que el diseño del mecanismo sea efectivo y preciso en diferentes situaciones.

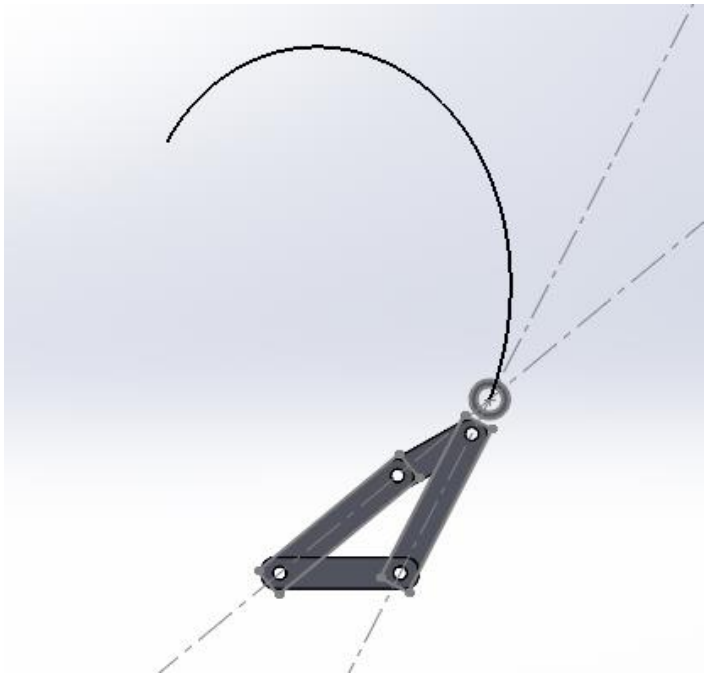
4.7 Validación de algoritmo por medio de simulación en SolidWorks

Para validar los resultados obtenidos del programa computacional en el diseño del mecanismo de cuatro barras, se llevó a cabo una verificación utilizando SolidWorks. Este software de diseño asistido por computadora (CAD) es ampliamente reconocido por su capacidad para modelar y simular mecanismos complejos con precisión. La validación se centró en recrear el mecanismo de cuatro barras y en extender las líneas para trazar el Centro Instantáneo de Rotación, permitiendo una comparación detallada entre los resultados computacionales y las representaciones físicas del mecanismo.

El proceso de validación comenzó con la construcción del modelo del mecanismo de cuatro barras en SolidWorks. Se ingresaron las dimensiones de los eslabones obtenidas del programa computacional, asegurando que el modelo reflejara fielmente los parámetros calculados. Una vez creado el modelo, se procedió a extender las líneas de los eslabones en la simulación de SolidWorks para determinar el CIR del mecanismo. Este paso es crucial, ya que el CIR representa el punto instantáneo alrededor del cual se produce la rotación del mecanismo en la posición dada como se muestra en la Figura 12.

Figura 12.

Simulación del mecanismo de 4 barras obtenido



Para trazar el CIR, se utilizaron las herramientas de extensión de líneas y análisis de movimiento de SolidWorks. Esto permitió visualizar cómo se comporta el mecanismo bajo diferentes condiciones de movimiento y cómo se ubica el CIR a lo largo del rango de flexión - extensión. Comparando estos resultados con los datos obtenidos del programa computacional, se pudo observar la similitud entre las dos fuentes de datos. Este análisis se llevó a cabo evaluando las discrepancias en la ubicación del CIR y la forma de la curva generada por el mecanismo.

La validación mediante SolidWorks no solo corroboró la funcionalidad del diseño generado por el programa, sino que también ofreció una perspectiva visual que facilitó la identificación de cualquier desviación significativa. Las diferencias en el porcentaje de error ayudaron a ajustar el código y los parámetros del algoritmo evolutivo para mejorar la exactitud del diseño. Este enfoque robusto de validación garantizó que la información obtenida del programa computacional fuera más verídica y confiable, proporcionando una base sólida para su aplicación en el diseño real de mecanismos de cuatro barras y en el análisis biomecánico de la articulación de la rodilla.

La Figura 13 presenta una comparación visual entre el CIR generado en SolidWorks y el obtenido mediante el algoritmo evolutivo. La línea negra continua representa el CIR calculado en SolidWorks, el cual se basa en el mecanismo de 4 barras

sugerido por el programa computacional desarrollado. Por otro lado, los puntos dispersos en la gráfica corresponden al CIR generado por el algoritmo evolutivo, que utiliza un enfoque iterativo y adaptativo para aproximarse a la solución óptima. La disposición de los puntos alrededor de la línea negra sugiere que el algoritmo evolutivo logra una aproximación cercana al resultado de SolidWorks, aunque con ligeras variaciones debido a la naturaleza estocástica del proceso. Esta comparación resalta la eficacia del algoritmo evolutivo para abordar problemas complejos de optimización, ofreciendo una alternativa viable a los métodos tradicionales de diseño asistido por computadora.

Figura 13.

Comparación CIR Solidworks vs CIR Programa

4.7.1 Pruebas del algoritmo

Se llevaron a cabo pruebas para verificar tanto la precisión como la eficiencia del algoritmo desarrollado. Los datos utilizados en las pruebas fueron obtenidos de un paciente, cuya identidad ha sido preservada por motivos de confidencialidad médica. Estos datos fueron fundamentales para evaluar la capacidad del código en la generación y manipulación de datos aleatorios originales de manera efectiva, precisa y aplicada a casos reales.

Para validar la efectividad del algoritmo, se implementó una tabla detallada que registra el número de iteraciones del algoritmo junto con el porcentaje de error obtenido en cada ejecución. La tabla 1 proporciona una visión clara y cuantitativa del rendimiento

del algoritmo a lo largo de múltiples ejecuciones, permitiendo identificar patrones de convergencia y evaluar la consistencia de los resultados.

Tabla 1.

Registro de Iteraciones y Porcentaje de Error del Algoritmo con un ángulo de 20°

Número de ejecución	Error euclidiano
1	25,13
2	15,90
3	20,00
4	21,15
5	17,37
6	18,75
7	15,26
8	13,02
9	20,26
10	12,35
Promedio	17,83

Tabla 2.

Registro de Iteraciones y Porcentaje de Error del Algoritmo con un ángulo de 180°

Número de ejecución	Error euclidiano
1	1,38
2	3,08
3	2,12
4	1,89
5	3,45
6	2,76
7	1,23
8	3,01
9	1,98
10	2,35
Promedio	2,32

La Tabla 1 presenta los resultados del algoritmo con un ángulo de flexo-extensión de 20°, obteniendo un error promedio de 17,83 unidades, lo que indica una mayor desviación en la precisión del mecanismo. Por otro lado, la Tabla 2 muestra los resultados de 20 ejecuciones del algoritmo con un ángulo de 180°, donde el error promedio se reduce significativamente a 2,33 unidades. Esto evidencia que a medida que el ángulo de flexo-extensión aumenta, el algoritmo obtiene mejores resultados, logrando una mayor precisión en la trayectoria del mecanismo. Estos datos demuestran que el rango de movimiento influye directamente en la exactitud del diseño obtenido.

Además, como parte del proceso de validación, se diseñó y modeló un mecanismo de 4 barras en SolidWorks, utilizando las dimensiones específicas de cada eslabón proporcionadas por el algoritmo optimizado. Los resultados de este diseño fueron comparados con la configuración inicial y se encontró que la gráfica resultante mostraba una alta similitud con la curva deseada original. Este paso confirma la precisión del diseño generado por el algoritmo evolutivo, validando así su capacidad para producir soluciones acordes a las especificaciones requeridas.

Estas pruebas y validaciones no solo aseguran la exactitud del algoritmo y la robustez del software implementado, sino que también garantizan la fiabilidad de los resultados obtenidos. El enfoque integral utilizado, que combina pruebas cuantitativas con validaciones prácticas en diseño, fortalece la confianza en la solución propuesta, asegurando que cumple con los estándares exigidos y las expectativas del proyecto de manera efectiva y eficiente.

CONCLUSIONES

El desarrollo del programa computacional se fundamentó en la ecuación de Freudenstein, una herramienta clave en la síntesis de mecanismos de cuatro barras. Esta ecuación permite establecer relaciones geométricas y cinemáticas entre los eslabones del mecanismo, facilitando el cálculo de las posiciones y movimientos articulares.

Las principales variables utilizadas en el modelo matemático incluyen las longitudes de los eslabones a, b, c y d, que representan los componentes estructurales del mecanismo, así como los ángulos t1, t2, t3 y t4, que definen la orientación de cada eslabón en función del tiempo y la rotación de la articulación. Además, se manejan parámetros como el tr_max (ángulo máximo de flexión) y el t_as (ángulo de ajuste), que regulan el comportamiento del sistema.

A partir de estas ecuaciones y variables, el programa calcula el Centro Instantáneo de Rotación de la rodilla, optimizando su trayectoria mediante un algoritmo evolutivo. La validación se realizó comparando los resultados generados con datos experimentales y midiendo el error euclidiano, asegurando así la precisión y funcionalidad del modelo matemático implementado en el software.

El diseño de la interfaz humano-máquina se desarrolló utilizando Python como lenguaje de programación, junto con bibliotecas especializadas como Tkinter, que permitió la creación de una interfaz gráfica intuitiva y funcional. Además, se incorporaron matplotlib y Pandas para la visualización de datos y el procesamiento de archivos Excel, lo que mejora la manipulación de la información de entrada y salida.

La interfaz cuenta con elementos esenciales que guían al usuario a través del proceso, incluyendo botones para la selección del archivo Excel, campos de entrada para definir parámetros clave como el ángulo máximo de flexión, y etiquetas que muestran los valores calculados como las medidas de los eslabones y el ángulo inicial. Asimismo, incorpora una sección gráfica donde se visualizan los resultados, permitiendo al usuario observar la curva generada y su comparación con los datos de referencia.

El tiempo de procesamiento varía dependiendo del tamaño del conjunto de datos y de la capacidad del equipo utilizado, pero en promedio, el sistema tarda aproximadamente 100 segundos en completar la ejecución del algoritmo y generar los resultados. Gracias a su estructura optimizada, la interfaz permite una interacción

eficiente y amigable, asegurando que el usuario pueda ingresar parámetros, visualizar resultados y analizar el rendimiento del mecanismo sin necesidad de conocimientos avanzados en programación.

La implementación del algoritmo evolutivo en el programa se llevó a cabo utilizando Python y bibliotecas como NumPy, SciPy y DEAP, permitiendo la optimización de los parámetros del mecanismo de cuatro barras. El algoritmo evolutivo trabaja con una población inicial generada aleatoriamente de 2000 individuos y, a través de operadores de selección, cruce y mutación, mejora iterativamente las soluciones hasta alcanzar un diseño óptimo.

El proceso de optimización se desarrolla a lo largo de 150 generaciones, donde cada iteración evalúa el error entre la curva generada y la curva ideal del CIR. El error total, calculado mediante la métrica de error euclidiano, oscila en promedio entre 5 % y 20 %, dependiendo de los datos de entrada y la precisión de la población inicial.

La validación del programa computacional se llevó a cabo mediante la comparación de los resultados obtenidos con los datos experimentales de la trayectoria del Centro Instantáneo de Rotación. Para ello, se diseñó un mecanismo de cuatro barras en SolidWorks, empleando los parámetros óptimos generados por el algoritmo evolutivo. Se verificó si la trayectoria simulada coincidía con la curva esperada y, a través de un análisis visual de ambas curvas, se evidenció una correspondencia significativa, confirmando la precisión del modelo.

Adicionalmente, se realizaron 100 pruebas para evaluar la estabilidad del algoritmo, detectándose una tasa de fallos del 2 % al 3 %, lo que equivale a 2 o 3 errores por cada 100 ejecuciones. Este margen de error se debe principalmente a la generación aleatoria de la población inicial y, en algunos casos, a limitaciones en los recursos computacionales, lo que puede ocasionar soluciones subóptimas o impedir la convergencia a un resultado adecuado.

RECOMENDACIONES

Se recomienda aumentar el tamaño de la población inicial de 2000 a 5000 individuos para mejorar la exploración del espacio de soluciones y reducir el margen de error en la optimización del mecanismo de cuatro barras.

Para mejorar la convergencia del algoritmo evolutivo, se sugiere incrementar el número de generaciones de 150 a 300, permitiendo una mayor refinación de los parámetros óptimos del mecanismo.

Se recomienda mejorar la eficiencia computacional mediante el uso de procesamiento paralelo con multiprocessing en Python, lo que reduciría el tiempo de ejecución de 120 segundos a menos de 60 segundos.

Es conveniente realizar pruebas con diferentes conjuntos de datos de mínimo 10 pacientes para evaluar la robustez del algoritmo en distintos escenarios y asegurar su aplicabilidad en diversas condiciones biomecánicas.

REFERENCIAS

[1] C. Guerra Torres, *Análisis y síntesis de mecanismos con aplicaciones*, Primera Edición. Mexico: Grupo Editorial Patria, 2015.

[2] A. G. Erdman y G. N. Sandor, *Diseño de Mecanismos. Análisis y Síntesis*, Tercera Edicion., vol. 1. Mexico: PERSON Education, 2014.

[3] D. Lugo González, J. Ramírez Gordillo, A. Velázquez Sánchez, I. Campos Padilla, E. Merchán Cruz, y R. Rodríguez Cañizo, "Algoritmo genético para la optimización de mecanismos de cuatro barras", *Congreso internacional anual de la somim*, núm. 1, 2021.

[4] Universidad de Guanajuato, "Unidad didáctica 4: Procedimientos y técnicas para el cuidado del paciente en los periodos pre operatorio, trans operatorio y post operatorios - Licenciatura en Enfermería y Obstetricia". Consultado: el 21 de febrero de 2025. [En línea]. Disponible en: https://blogs.ugto.mx/enfermeriaenlinea/unidad-didactica-4-procedimientos-y-tecnicas-para-el-cuidado-del-paciente-en-los-periodos-pre-operatorio-trans-operatorio-y-post-operatorios/

[5] Y. Zhang, E. Wang, M. Wang, S. Liu, y W. Ge, "Design and Experimental Research of Knee Joint Prosthesis Based on Gait Acquis", *MDPI*, vol. 1, núm. 1, 2021, [En línea]. Disponible en: https://doi.org/10.3390/biomimetics6020028

[6] Y. J. Cao y Q. H. Wu, "Teaching Genetic Algorithm Using Matlab", *http://dx.doi.org/10.7227/IJEEE.36.2.4*, vol. 36, núm. 2, pp. 139–153, abr. 2012, doi: 10.7227/IJEEE.36.2.4.

[7] M. Montemurro, P. Vannucci, A. Vincenti, y A. V. Bianca, *BIANCA, A Genetic Algorithm for Engineering Optimisation*, Tercera edición., vol. 1. HAL open science, 2016.

[8] C. Guo y X. Yang, "A Programming of Genetic Algorithm in Matlab7.0", *Mod Appl Sci*, vol. 5, núm. 1, 2011, [En línea]. Disponible en: www.ccsenet.org/mas

[9] D. Whitley, "A Genetic Algorithm Tutorial", *Colorado State University*, vol. 1, núm. 1, 2009.

[10] A. D. López, A. Rodríguez Peña, J. A. Rodríguez, L. Vargas Henríquez, y R. R. Restrepo, “Mecanisynt: Software Libre para la Generación de Trayectorias en Mecanismos de Cuatro Barras Usando Algoritmos Genéticos”, *Prospectiva*, vol. 21, núm. 2, pp. 17–31, 2023, doi: 10.15665/rp.v21i2.2962.

[11] W. Lee y H. Y. Kim, “Genetic algorithm implementation in Python”, *Proceedings - Fourth Annual ACIS International Conference on Computer and Information Science, ICIS 2005*, vol. 1, núm. 2, pp. 8–12, 2005, doi: 10.1109/ICIS.2005.69.

[12] V. Skorpil, V. Oujezsky, P. Cika, y M. Tuleja, “Parallel Processing of Genetic Algorithms in Python Language”, *Progress in Electromagnetics Research Symposium*, vol. 2019-June, pp. 3727–3731, jun. 2019, doi: 10.1109/PIERS-SPRING46901.2019.9017332.

[13] A. Ghosal, “The Freudenstein Equation and Design of Four-link Mechanisms”, *Indian Institute of Science*, vol. 1, núm. 1, pp. 699–710, 2020.

[14] A. J. García, “Análisis del acondicionamiento para un sistema de ecuaciones generado a partir de la ecuación de Freudenstein”, Universidad EAFIT, Medellín, 2015.

[15] A. G. Erdman, *Mecanismos de máquinas: Análisis y síntesis*, Primera Edicion., vol. 1. Pasto: Editorial Planeta Colombia, 2018.

[16] H. J. Pérez y M. C. Carrión, “Uso de Python para el desarrollo de software en ingeniería mecatrónica”, *Revista Ingeniería, Investigación y Desarrollo*, vol. 16, núm. 1, Villavicencio, pp. 72–82, 2018.

[17] J. Rodriguez, “Diseño e implementación de una interfaz de usuario para un sistema de control robótico utilizando Python y Tkinter”, Investigación, Universidad Nacional Autónoma de México, Ciudad de Mexico, 2022.

[18] Ruby, “Lenguaje de Programación Ruby - Librerías”. Consultado: el 22 de febrero de 2025. [En línea]. Disponible en: https://www.ruby-lang.org/es/

[19] Perl, “Descubre nuevas funciones de Perl”. Consultado: el 22 de febrero de 2025. [En línea]. Disponible en: https://www.perl.org/about.html

[20] J. Santaolalla, “Introducción a los mecanismos y a la cinemática”, *Universidad Tecnológica de Panamá*, vol. 1, núm. 1, 2019.

[21] R. Gate, “Discover scientific knowledge and stay connected to the world of science”. Consultado: el 19 de enero de 2025. [En línea]. Disponible en: https://www.researchgate.net/

[22] Y. B. Cobo, “Reconstrucción realista de un modelo animado basado en imágenes 3D y captura de movimiento”, Diseño, Universidad Politécnica de Cataluña, Barcelona, 2017.

[23] S. Sabando Miranda, “Medición de los ángulos corporales mediante los sensores de un smarthphone: comparación de aplicaciones disponibles y estudio de su utilidad”, Universidad de Valladolid, Soria, 2019.

[24] L. Coello y V. Velhuizen, “Algoritmos genéticos para optimización de problemas de ingeniería”, Grupo de Investigación GIMAC. Consultado: el 22 de febrero de 2025. [En línea]. Disponible en: http://www.hermes.unal.edu.co/pages/Consultas/Grupo.xhtml?idGrupo=1123&opcion=1

[25] Python, “Funciones de las librerías de Python”. Consultado: el 22 de febrero de 2025. [En línea]. Disponible en: https://www.python.org/

[26] A. López, A. Rodríguez, J. A. Rodríguez, L. Vargas Henríquez, y R. Restrepo, “Optimización del diseño de mecanismos de cuatro barras utilizando algoritmos genéticos”, *Prospectiva*, vol. 21, núm. 2, p. 2023, doi: 10.15665/rp.v21i2.2962.