ICYTE
Instituto de investigaciones científicas y tecnológicas en electrónica

Registración y fusión de imágenes médicas mediante técnicas de optimización estadística e inteligencia computacional

Tesis

Presentada en la *Facultad de Ingeniería* de la *Universidad Nacional de Mar del Plata*, en consideración para obtener el grado académico de

Doctor en Ingeniería orientación Electrónica

Por

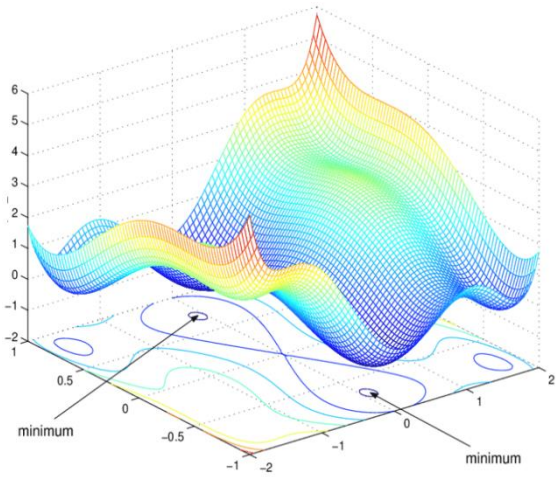
Ing. Ramiro Fernando Isa Jara

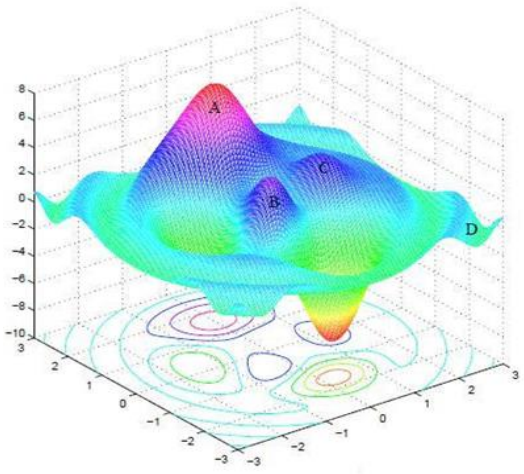
Dirigida por

Directora: Dra. Ing. Virginia Laura Ballarin

Codirector: Dr. Ing. Gustavo Javier Meschino

Codirector de beca: Ing. Hugo Moreno Avilés PhD





Mar del Plata, Argentina

2019

A mis amados padres Efraín y Laura, a mis hermanos, a mi angelito, a Johan Q.I. y a la memoria de mis seres queridos que ya no están.

Palabras de agradecimiento

El doctorado ha sido mi más grande desafío hasta ahora, pero también uno de mis sueños anhelados. Siempre supe que los sueños demandan mucha dedicación, esfuerzo y constancia, lo cual da sentido y propósito a la vida. El presente documento es la culminación de este camino que empezó 5 años atrás, los cuales puedo decir con toda certeza, han sido hermosos por todo el aprendizaje a nivel profesional, pero sobre todo a nivel personal. Llegar a este momento también significa representar a todas las personas que han estado a mi lado, desde mucho antes y durante toda esta etapa, expresándome su cariño, comprensión, motivación y apoyo de distintas maneras. A todas ustedes les dedico estas líneas.

Con todo mi amor, quiero agradecer a mis padres Efraín Isa y Laura Jara por haberme dado sus mejores años de vida, por amarme, por cuidarme, motivarme y guiarme para cumplir mis sueños y tener una vida feliz. He sentido profundamente el amor de Uds. hacia mí, aunque, ha sido una larga travesía, pero ha sido muy gratificante cada día y cada minuto, por lo cual, no puedo más que sentirme honrado de ser uno de sus hijos. A mis hermanos queridos Mónica, Jenny y Franklin, quienes me han brindado todo su apoyo y cariño, especialmente estos últimos años que he pasado fuera de casa. Quiero que sepan lo mucho que los quiero. A Johan, mi pequeño sobrino, quien me transmite su amor incondicional y a quien he tenido siempre en mi corazón. Y al resto de mi familia, quienes han estado pendientes de mí transmitiéndome todo su apoyo y cariño para que pueda culminar mi meta.

Mi agradecimiento profundo a la Argentina y a la hermosa Mar del Plata por ser mi segundo hogar. Inmensa gratitud al Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET) por el financiamiento otorgado durante estos años para culminar mi investigación y a la Facultad de Ingeniería de la Universidad Nacional de Mar del Plata por recibirme y formarme como investigador, para estar a la vanguardia de las nuevas herramientas tecnológicas y así poder aportar al crecimiento de nuestra sociedad.

A mis directores Dra. Virginia Ballarin y Dr. Gustavo Meschino, a quienes con todo cariño les digo infinitas gracias por abrirme las puertas de su hermoso país y por su confianza para emprender este camino. Durante este tiempo he aprendido de sus conocimientos y experiencias en el campo científico y también en el personal. Al Dr. Hugo Moreno A. quien fue mi profesor de grado y mentor, a quien le agradezco mucho su tiempo y colaboración para poder emprender mi carrera de postgrado. Admiro mucho su capacidad y don de gente, lo cual me motiva a seguir aprendiendo cada día.

De igual manera, un agradecimiento inmenso a mi Comisión de Seguimiento conformada por la Dra. Adriana Scandurra, el Dr. Juan Ignacio Pastore y el Dr. Juan Pablo Graffigna. Gracias por todo su tiempo, ayuda y por sus sugerencias durante el doctorado. A los miembros del jurado conformado por la Dra. Mariana del Fresno, el Dr. Leonardo Arnone y el Dr. Claudio Delreiux, muchas gracias por su tiempo para la revisión de esta Tesis y por sus recomendaciones, las cuales han sido valiosas para le mejora en el contenido, presentación y formalidad de este documento.

A todos mis compañeros y amigos que son parte del Laboratorio de Procesamiento de imágenes, gracias por su amistad y aprecio. Ha sido un gusto inmenso conocerlos y compartir con ustedes el día a día. Mi deseo grande es que sigan cumpliendo cada uno de sus sueños.

A Stefy Cujano y Fran Buchelly, amigos queridos gracias por cada uno de los instantes compartidos, por las risas, por las aventuras, por las confidencias, por el apoyo y por su don de familia. Quiero que sepan que los quiero mucho y que las anécdotas que vivimos siempre estarán en mi mente y corazón.

A todas y cada una de las personas que pude conocer y con quienes compartí tiempo y amistad, gracias por haberme enseñado a vivir más libre. A mi grupo de amigos cercano, gracias por todo su cariño, por su sinceridad, por su aprecio. A todos los llevaré en mis más bonitos recuerdos.

Y finalmente, agradecer con todo mi Ser, a la Vida, a esa inteligencia infinita que ha guiado cada una de las experiencias que he vivido y que me han llevado a la conexión profunda con el Amor. Gracias por haberme dado la oportunidad de reencontrarme conmigo y poder hoy vivir con más alegría, con más libertad, con más amor hacia mí y hacia todos. Gracias infinitas Universo.

Ahora empieza una nueva etapa en mi vida, pero quiero que sepan lo agradecido que estoy con todos y con Todo.

Ramiro Fernando Isa Jara

Mar del Plata, 5 de noviembre de 2019

De nuestros miedos
nacen nuestros corajes
y en nuestras dudas
viven nuestras certezas.

Los sueños anuncian
otra realidad posible
y los delirios otra razón.

En los extravíos
nos esperan hallazgos,
porque es preciso perderse
para volver a encontrarse.

Eduardo Galeano

Índice general

Índice de figuras

Índice de tablas

Acrónimos

PDI	Procesamiento digital de imágenes	
IRM	Imágenes de resonancia magnética	
TAC	Tomografía axial computada	
PD	*Proton density*	Densidad de protones
fMRI	*Functional magnetic resonance imaging*	Resonancia magnética funcional
SPECT	*Single-photon emission computed tomography*	Tomografía computarizada de emisión monofotónica
PET	*Positron-emission tomography*	Tomografía por emisión de positrones
MI	*Mutual Information*	Información mutua
CC	*Correlation Coefficient*	Coeficiente de correlación
PDF	*Probability Density Function*	Función de densidad de probabilidad
PCA	*Principal Component Analysis*	Análisis de componentes principales
ICA	*Individual Component Analysis*	Análisis de componentes individuales
SIFT	*Scale – Invariant Feature Transform*	Transformación de características invariantes a la escala
SURF	*Speeded – Up Robust Features*	Características robustas aceleradas
RANSAC	*Random Sample Consensus*	Algoritmo de muestra aleatoria consensuada
SI	*Swarm Intelligence*	Inteligencia de enjambres
PSO	*Particle Swarm Optimization*	Optimización por enjambre de partículas
ACO	*Ant Colony Optimization*	Optimización por colonia de hormigas
FWA	*Fireworks Algorithm*	Algoritmo de fuegos artificiales

ABC	*Artificial Bee Colony*	Optimización por colonia de abejas
LiA	*Lightning Algorithm*	Algoritmo de rayos
AI	*Artificial Intelligence*	Inteligencia artificial
CI	*Computational Intelligence*	Inteligencia computacional
ANN	*Artificial Neural Networks*	Redes neuronales artificiales
FS	*Fuzzy Systems*	Sistemas difusos
EC	*Evolutionary computation*	Computación evolutiva
RS	*Random Search*	Búsqueda aleatoria
SA	*Simulated annealing*	Recocido simulado
SVM	*Support Vector Machines*	Máquinas de soporte vectorial
RL	*Reinforcement Learning*	Aprendizaje por refuerzo
MDPs	*Markov Decision Process*	Procesos de decisión de Markov
DRL	*Deep Reinforcement Learning*	Aprendizaje por refuerzo profundo

Resumen

En el presente trabajo se aborda la temática de la *Registración de imágenes médicas*. La registración tiene como finalidad alinear geométricamente dos o más imágenes que comparten información común y, a su vez, presentan información complementaria. Dentro del campo médico, este proceso tiene gran relevancia, ya que permite la comparación de imágenes que han sido adquiridas en distintos instantes de tiempo o en distinta modalidad. Además, es el paso previo para procesos mucho más complejos como son la fusión de imágenes, la segmentación, entre otros.

La registración es un problema de optimización que incluye el procesamiento de imágenes. Por tanto, a lo largo de este trabajo se abordan estos dos temas centrales desde varias perspectivas. El caso de la optimización se aborda desde el enfoque estocástico y de la programación evolutiva mediante el estudio y posterior aplicación de algoritmos inspirados en el comportamiento de sistemas biológicos entre los que están las colmenas de abejas o las colonias de hormigas.

El procesamiento de imágenes es, sin duda, una de las técnicas actuales más relevantes, debido a la alta capacidad de los equipos de adquisición y de las herramientas computacionales para su posterior adecuación y mejora. Desde esta perspectiva, se ha abordado el filtrado y extracción de características invariantes en imágenes. Esta extracción permite obtener información relevante tanto local como global de una imagen, pero además con la propiedad de invariancia frente a ciertas transformaciones geométricas, lo que permite el emparejamiento de dos o más imágenes con información compartida. Los métodos SIFT y SURF han servido como base para el desarrollo de una propuesta para la extracción de forma automática de marcadores anatómicos en imágenes médicas y su posterior aplicación en el proceso de registración.

Una de las áreas más atractivas en los últimos años dentro de las ciencias computacionales y la ingeniería, es sin duda la inteligencia computacional, campo fundamental de la inteligencia artificial. Esta depende directamente de los métodos que permiten que un sistema o agente inteligente pueda aprender a partir de datos. A esto se lo conoce como Aprendizaje de máquina o *Machine Learning*. Este trabajo aborda uno de sus métodos principales denominado *aprendizaje por refuerzo.* Además, se aborda el aprendizaje profundo o *Deep Learning* desde el fundamento del aprendizaje por refuerzo. Este tiene como finalidad que un agente artificial aprenda desde la experiencia al interactuar directamente con su entorno.

El algoritmo *Q-Learning* ha servido de base para el estudio de este método junto con los nuevos aportes basados en sistemas no lineales como las redes neuronales que han dado el inicio a una amplia gama de aplicaciones en teoría de juegos, redes de energía inteligentes y, por supuesto, el diagnóstico médico. En este trabajo se aborda el aprendizaje por refuerzo y el aprendizaje por refuerzo profundo enfocados en dar una alternativa a la registración de imágenes utilizando esta tecnología y métodos muy recientes.

Esta tesis tiene como objetivo central brindar una nueva visión con relación a la registración de imágenes mediante el aporte de nuevos resultados obtenidos con los métodos mencionados previamente. Estos han demostrado ser interesantes y prometedores, especialmente cuando se los compara con métodos muy aceptados en este campo de investigación.

Esta Tesis está estructurada de la siguiente manera:

- **Capítulo 1 – Introducción**
  En este capítulo se presenta el contexto general y específico en que están involucrados la registración de imágenes. Además, se analizan los avances previos realizados, especialmente dentro del área médica. Finalmente, se expone la metodología que será utilizada en el desarrollo de esta tesis.
- **Capítulo 2 – Registración y fusión de imágenes**
  En este capítulo se presentan los conceptos y características propias de la registración y la fusión de imágenes. También se introducen las técnicas de procesamiento de imágenes más útiles y adecuadas para la solución al problema de alineamiento.
- **Capítulo 3 – Optimización**
  En este capítulo se presenta la definición formal y los alcances de la optimización. Además, se introduce en el ámbito de la optimización estocástica junto con los métodos y técnicas más utilizadas. Finalmente, se presentan los aportes realizados en torno a la optimización estocástica y posteriormente los resultados dentro de la registración de imágenes médicas.
- **Capítulo 4 – Extracción de características**
  En este capítulo se analizan y describen de manera específica las técnicas de extracción automática de características en una imagen. Se describen algunos métodos utilizados con este objetivo. Finalmente, se presentan los aportes realizados utilizando este método en torno a la registración de imágenes médicas.
- **Capítulo 5 - Inteligencia computacional**
  En este capítulo se presenta el paradigma de la inteligencia computacional y su campo de aplicación dentro del procesamiento de imágenes. Además, se describe el enfoque del aprendizaje por refuerzo y del aprendizaje por refuerzo profundo, el cual se considera uno de los adecuados para abordar la registración de imágenes. Finalmente, se presentan los aportes realizados a esta área con imágenes médicas.
- **Capítulo 6 – Conclusiones**
  En este capítulo se exponen las conclusiones obtenidas durante el presente trabajo. Además, se analizan los alcances de los objetivos propuestos y se propone el trabajo futuro.

Capítulo 1 Introducción

> Equipado con sus cinco sentidos, el hombre explora el Universo que lo rodea y a sus aventuras las llama Ciencia.
>
> *Edwin Powell Hubble*

El procesamiento digital de imágenes es un área de investigación que ha cobrado mucho interés en las últimas décadas debido a la amplia variedad de potenciales aplicaciones que pueden ser desarrolladas. Además, el PDI tiene su base en áreas fundamentales como la matemática, la geometría, las ciencias computacionales entre otras varias. Esto junto con los resultados obtenidos hasta el momento, los cuales son de fácil interpretación a nivel humano, la hacen un área cada vez más atractiva y con un amplio horizonte de desarrollo.

1.1 Contexto específico: Procesamiento de imágenes

El PDI es una subárea del procesamiento de señales que tiene como objeto la manipulación de una imagen digital por medio de tecnología y equipos informáticos para la graficación y el procesamiento de información [1]. La visión es uno de los sentidos más preciados para los seres humanos puesto que los permite percibir su entorno. Las imágenes y la computadora dieron lugar a las imágenes digitales, la cuales han permitido mejorar la capacidad para ver, percibir y entender entornos que normalmente están fuera de su alcance, ya sea por la franja infrarroja o por la distancia a la que se encuentran. En la actualidad debido al avance investigativo y tecnológico, los procesos de análisis y extracción de información de imágenes son cada vez más robustos y autónomos superando, en ciertas tareas específicas, la capacidad humana.

Una de las áreas con mayor impacto humano y social es la medicina, la cual se ha visto ampliamente beneficiada por estos avances tecnológicos. El equipamiento médico ha permitido que profesionales de esta área mejoren su capacidad en cuanto al diagnóstico temprano de enfermedades, en las intervenciones quirúrgicas y el posterior seguimiento del tratamiento de un paciente debido a la amplia cantidad de información disponible para validar datos y resultados. Esto último hace indispensable la generación de nuevos métodos, algoritmos y tecnología capaces de solucionar los nuevos y crecientes desafíos que pudieran presentarse.

El diagnóstico por imágenes, conocido en el campo médico como *Imagenología*, permite obtener información del cuerpo humano para realizar estudios cada vez más complejos de la anatomía y su funcionamiento. Esta información, que se muestra en imágenes, proviene de equipos con altos estándares de funcionamiento basados generalmente en radiación controlada de energía, como los Rayos X para detectar tejido óseo o el ultrasonido para análisis fetal. Estas imágenes son de alta resolución y pueden ser captadas en distintas modalidades, además de las ya mencionadas, como son las IRM, las imágenes de TAC entre otras. Estas imágenes contienen gran cantidad de información que el especialista deberá interpretar correctamente para un buen diagnóstico y posterior tratamiento.

El procesamiento de imágenes incluye: el filtrado y acondicionamiento, la segmentación automática o semiautomática, reconocimiento de patrones, el seguimiento de objetos, la registración de imágenes, el análisis y la comparación de estructuras fisiológicas, entre otras tareas. Una de las áreas que genera gran interés, actualmente, es la Visión por computadora. En ésta, se pretende emular la capacidad del procesamiento visual humano en un sistema computacional para abordar procesos y tareas cada vez más complejas como el reconocimiento patrones o la identificación de características. Para ello se cuenta con algoritmos cada vez más rápidos y con un alto rendimiento capaces de realizar tareas con gran exactitud, lo cual ha permitido el reconocimiento óptico de caracteres, la reconstrucción de modelos 3D, las capturas y análisis de movimiento, el reconocimiento de rostros, la delimitación de regiones de interés, entre otros logros [2].

1.2 Contexto específico: Registración de imágenes

La registración es un paso previo al procesamiento de imágenes que tiene como objetivo la alineación geométrica de dos o más imágenes de la misma región de interés que han sido capturadas en distintos instantes de tiempo o en distinta modalidad. La registración es inevitable en los procesos de comparación de información, teledetección, reconstrucciones y mosaicos de imágenes, detección de cambios o la restauración de imágenes multicanales [3].

En el caso clínico, las imágenes tienen un rol importante previo al diagnóstico. Las imágenes médicas pueden ser de tipo anatómico o funcional[1], donde cada una de ellas aporta información complementaria, que permite ampliar la perspectiva y mejorar el análisis del caso en estudio. Para ello, las imágenes deben estar correctamente alineadas. Esto permite realizar el seguimiento del tratamiento de un paciente, la caracterización de una determinada patología y la combinación o fusión de información de una o varias imágenes en distintas modalidades, por ejemplo, en la fusión de una imagen MRI con una CT adquiridas en un mismo paciente.

En la literatura disponible se pueden revisar los aportes realizados para la solución de esta problemática, lo cual ha permitido el desarrollado de algoritmos semiautomáticos y automáticos para el alineamiento preciso de imágenes. Estos algoritmos, al ser robustos y con un alto rendimiento, hacen cada vez menos necesaria la intervención humana en determinados casos. Sin embargo, es imposible desarrollar un método de registración general debido a las características particulares de las imágenes involucradas, además de las deformaciones que pudieran presentarse en cada caso de estudio.

En el proceso de registración siempre se tendrá la imagen principal denominada *imagen referencia o base* y la imagen a registrar denominada *imagen móvil o deformada.* El objetivo es encontrar una transformación geométrica que mapee la información de la imagen móvil para llevarla al espacio geométrico de la imagen de referencia. Para esto se tiene en cuenta ciertos parámetros que categorizan una registración, entre las cuales están: la dimensionalidad, la naturaleza de la registración, la naturaleza de la transformación, el dominio de la transformación, la interacción, el procedimiento de optimización, las modalidades involucradas, el sujeto y el objeto [4].

En la **Fig. 1.1**, se muestra el esquema de la registración de imágenes donde se detallan, de manera general, lo procesos involucrados previa la obtención de la imagen final o registrada.

[1] Las imágenes anatómicas aportan información del estado corporal, muscular y óseo, mientras que las funcionales aportan el estado del metabolismo y funcionamiento de órganos.

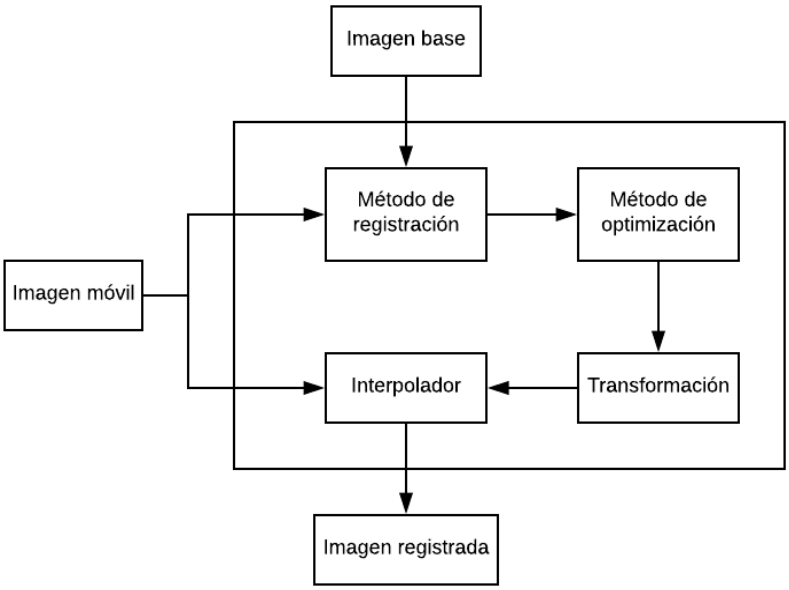


Fig. 1.1: Descripción general de los procesos que intervienen en la registración de imágenes.

En esta investigación, se han utilizado y propuesto técnicas de procesamiento de imágenes que permitan mejorar los resultados obtenidos en la registración de imágenes.

- ***Extracción de características invariantes:*** Mediante la utilización de filtros y modelos matemáticos, se pueden extraer características (features) de una imagen. Una característica es un punto o una zona distintiva, por ejemplo, una esquina, un borde o la región de un objeto dentro de la imagen. La propiedad principal de estas características es que son invariantes al cambio de escala, rotación o traslación de dicho objeto, incluso pueden ser robustas al cambio de iluminación y contraste. En el caso de la registración, las características son extraídas de la imagen de referencia y de la imagen móvil, para su comparación, emparejamiento y finalmente el cálculo de la transformación geométrica.

1.3 Contexto específico: Inteligencia computacional

La inteligencia computacional, componente fundamental de la más genérica inteligencia artificial, desde sus inicios ha despertado el interés de muchos investigadores, con el objetivo de crear y diseñar sistemas artificiales "inteligentes". Estos sistemas, en general, han nacido de la inspiración sobre el comportamiento y adaptación de los sistemas biológicos en la naturaleza. Las aplicaciones para estos sistemas están en varias áreas de investigación, por ejemplo, la optimización estocástica, el aprendizaje por refuerzo, el aprendizaje supervisado y no supervisado, entre otros.

Como se mencionó en la sección anterior, la visión humana es uno de los temas de investigación más apasionantes para la inteligencia computacional. Es por ello, que el procesamiento de imágenes se ha vinculado directamente con el uso de algoritmos "inteligentes". Sin bien, aunque persiste el debate sobre el nivel de "inteligencia" que presentan los agentes artificiales, es imposible negar los increíbles resultados que se han obtenido en términos de robustez, generalización y autonomía frente a tareas de alta complejidad.

Actualmente, el aprendizaje máquina o *Machine Learning* es la nueva tendencia para la inteligencia computacional. Una subárea de esta conocida como *Deep Learning* o aprendizaje

profundo también ha mostrado su amplio rango de aplicaciones con resultados exitosos. Las librerías de código abierto para Deep Learning, liberadas por Google a través de su empresa *Deep Mind*, ha permitido que muchas más personas se involucren de distintas maneras a este apasionante campo de investigación.

En esta investigación se ha abordado varias técnicas de inteligencia computacional, teniendo como finalidad principal el uso y la propuesta de métodos que ayuden a resolver la problemática de la registración de imágenes.

- ***Optimización estocástica:*** La optimización es una rama de las matemáticas que tiene como finalidad encontrar el mejor resultado a un determinado problema bajo ciertas restricciones. En la optimización estocástica se busca el mejor resultado utilizando algoritmos "inteligentes" basados en el comportamiento de varios sistemas biológicos, entre los que están, la genética, las aves, las abejas, las hormigas, etc.
- ***Aprendizaje por refuerzo:*** Este es uno de los tres métodos principales que componen el aprendizaje de máquina, inspirado en el aprendizaje humano. La idea principal es tener un agente que interaccione con su entorno, con la finalidad de que "aprenda" la política para la selección de las mejores acciones que le permitan cumplir un objetivo propuesto. Las aplicaciones de este paradigma están enfocadas generalmente a juegos; sin embargo, ha sido muy útil utilizarlo con esta finalidad. Además, el aprendizaje por refuerzo puede utilizar también redes neuronales para mejorar su rendimiento, lo que permite abordar el aprendizaje profundo desde esta perspectiva.

1.4 Contexto general

Al ser la registración un proceso importante para la comparación y análisis de información entre dos o más imágenes, sus métodos están ampliamente desarrollados para varias áreas, como son: las ciencias médicas, las imágenes satelitales y aeroespaciales, la visión robótica, la visión por computadora, la teledetección entre otras. Cada una de estas áreas cuenta con protocolos propios para la adquisición y almacenamiento de imágenes acorde a sus características y propiedades.

Los aportes, que se presentan a continuación, han servido como base para encarar la registración de imágenes. Estos han sido elegidos por las distintas perspectivas con las que abordan esta temática. R. Suganya et al. en [5], por ejemplo, propone un método para la registración de imágenes médicas de cráneo, utilizando la información de intensidad de sus píxeles y utiliza la información mutua (MI) como medida de similaridad. El método realiza un alineamiento inicial con los centros de masa y la registración está basada en el cálculo de la suma de la proyección de intensidades como método iterativo. Esta propuesta ha sido probada en registración multimodal entre imágenes de resonancia magnética MRI y tomografía computada CT.

Q. Li et al. en [6] propone la registración utilizando el algoritmo de enjambres PSO como método de optimización global para transformaciones rígidas. Las pruebas se realizan con imágenes de RM y PD craneales y con imágenes satelitales en distintas modalidades. T. Lin et al. en [7] propone la registración de imágenes médicas utilizando ACO . En este trabajo, los píxeles de las imágenes son tratados como nidos de hormigas y se define como medidas de similaridad el coeficiente de correlación (CC) y la suma de las diferencias cuadráticas (SSD). El valor SSD representa el "alimento" para las hormigas, durante las iteraciones y el algoritmo reduce esta diferencia por tanto el valor de correlación entre las imágenes aumenta. Las pruebas fueron realizadas con imágenes de MRI cerebrales con deformaciones rígidas y MRI de rodilla para deformaciones no lineales.

K. Krish et al. en [8] propone un método de registración aplicado en imágenes satelitales, basado en la extracción de características invariantes por medio del detector Harris-Laplaciano. La función objetivo se calcula por medio de Regresión por mínimos cuadrados no lineal (NLLS), ajustada mediante RANSAC para obtener la mejor transformación descartando los puntos aislados. Y. Chen et al. en [9] propone un algoritmo de PSO híbrido, ya que utiliza varios conceptos de los algoritmos genéticos. Se propone realizar subpoblaciones dentro del enjambre, cada una de ellas obtiene su mejor individuo mediante cruces y mutaciones. La registración se lleva a cabo por medio de la información mutua como medida de similaridad y está aplicada en imágenes de TAC y RM.

Es a partir del desarrollo de algoritmos de optimización global y de extracción de características como se ha mejorado el rendimiento y los resultados para la registración de imágenes, especialmente en el caso de las imágenes médicas. Debido a las capacidades computacionales disponibles actualmente, se hace factible abordar problemas cuyo espacio de soluciones es NP (no polinomiales), es decir, no solucionables en tiempo polinomial. Una de las formas óptimas de hacerlo es mediante el uso e implementación de algoritmos estocásticos como los que han sido mencionados.

1.5 Objetivos y aportes de esta Tesis

La registración es, en esencia, un problema de optimización donde se busca calcular la transformación geométrica óptima que disminuya la diferencia entre la imagen de referencia y la móvil. Este proceso debe tener en cuenta el uso adecuado de los recursos computacionales disponibles junto con un tiempo razonable para el procesamiento.

En este sentido, la finalidad principal de esta Tesis es abordar la inteligencia computacional y el procesamiento de imágenes desde dos perspectivas, la primera relacionada con la optimización estocástica y la segunda con los paradigmas de aprendizaje computacional. Con base en esto, la hipótesis planteada para el desarrollo de esta investigación es:

Las técnicas de optimización estadística y de inteligencia computacional aplicadas en el contexto de la registración de imágenes aportarán avances significativos.

Para esto se propone como objetivo general de esta Tesis:

- Adquirir nuevos conocimientos de algoritmos evolutivos y aplicarlos a registración y fusión de imágenes, de manera interdisciplinaria, brindando servicios a la comunidad científica y de allí a la comunidad en general.

Y los objetivos específicos comprenden:

- Estudiar la optimización de algoritmos basada en técnicas de algoritmos evolutivos, en particular del enfoque de la inteligencia de enjambres
- Diseñar un método de registro automático basado en algoritmos evolutivos, con la menor intervención posible de un usuario experto.
- Implementar la fusión de las imágenes registradas, combinando en una única imagen sintética información anatómica, funcional y metabólica de pacientes en estudio.
- Facilitar la interpretación de la información visual mediante la utilización de las imágenes combinadas, evitando imprecisiones anatómico-espaciales.
- Aplicar los algoritmos desarrollados en imágenes simuladas y reales.

- Implementar los algoritmos propuestos en un lenguaje de alto nivel, apto para prototipos, como por ejemplo MATLAB®.
- Obtener conclusiones de las distintas técnicas estudiadas.
- Presentar conclusiones en congresos y publicaciones de las temáticas involucradas.
- Trabajar en conjunto con profesionales informáticos en proyectos del área de procesamiento de imágenes digitales.
- Desarrollar un léxico compatible con posibles profesionales médicos que intervendrían en la evaluación de resultados.

Por tanto, se propone la aplicación de varios algoritmos de inteligencia de enjambres que no han sido utilizados previamente en registración. Además, se ha realizado una propuesta en relación con la extracción de características en imágenes utilizando el filtro de Gabor bajo una perspectiva novedosa y finalmente se ha aplicado los algoritmos de aprendizaje por refuerzo y aprendizaje profundo, donde se ha realizado pequeños aportes, para mejorar los resultados obtenidos.

En todos los casos abordados se pretende minimizar la intervención humana durante el proceso de la registración de imágenes.

1.6 Publicaciones que sustentan la Tesis

En las siguientes publicaciones, se han realizado aportes en torno a la registración de imágenes médicas. Esta Tesis está sustentada en ellas.

- **Proceedings**

  Isa-Jara Ramiro, Buchelly F.J., Meschino G.J., Ballarin V.L. **(2016)** "*Improved Particle Swarm Optimization algorithm applied to rigid registration in medical images".* In: Torres I., Bustamante J., Sierra D. (eds) VII Latin American Congress on Biomedical Engineering CLAIB 2016, Bucaramanga, Santander, Colombia, October 26th -28th, 2016. IFMBE Proceedings, vol 60. Springer, Singapore.

  Isa-Jara Ramiro, Meschino Gustavo, Ballarin Virginia. **(2017)** *"Identification of invariant anatomical markers for medical image registration using Gabor Filters",* XXI Congreso Argentino de Bioingeniería - SABI 2017, ISBN: 978-950-33-1406-7. Córdoba-Argentina.

  Isa-Jara Ramiro, Meschino Gustavo, Ballarin Virginia. **(2019)** *"A comparative study of Reinforcement learning algorithms applied to medical image registration",* Paper accepted in 2019 International Conference in Biomedical and Health Informatics. Taipei-Taiwan. Published in: VIII Latin American Conference on Biomedical Engineering and XLII National Conference on Biomedical Engineering, Cancún, Mexico.

  Buchelly, F.J., Isa-Jara, Ramiro F., Zalazar, L., Cesari, A., Pastore, J.I., Ballarin, V. **(2019)**. *"Comparative analysis of different techniques to determine motility parameters in video sequences of ram and buck sperm*". Published in: VIII Latin American Conference on Biomedical Engineering and XLII National Conference on Biomedical Engineering. Cancún, Mexico.

- **Competencias**

  Isa-Jara Ramiro, Meschino Gustavo., Ballarin Virginia. **(2018)** *"Comparación de algoritmos de inteligencia por refuerzo aplicados a la registración de imágenes médicas.",* **Primer lugar** en el IEEE – TRIC VIII Torneo regional de inteligencia computacional. Ciudad Autónoma de Buenos Aires - Argentina

Referencias

[1] R. C. Gonzalez and R. E. Woods, "Digital Image Processing," 2008.

[2] R. Szeliski, "Computer Vision: Algorithms and Applications," 2010.

[3] B. Zitová and J. Flusser, "Image registration methods: a survey," *Image Vis. Comput.*, vol. 21, pp. 977–1000, 2003.

[4] J. B. A. Maintz and M. A. Viergever, "A Survey of Medical Image Registration," *Med. Image Anal.*, vol. 2, no. 1, pp. 1–37, 1998.

[5] R. S. Suganya R., Priyadharsini K., "Intensity Based Image Registration by Maximization of Mutual Information," *Int. J. Comput. Apl.*, vol. 1, no. 20, 2010.

[6] Q. Li and I. Sato, "Multimodality Image Registration by Particle Swarm Optimization of Mutual Information," *LNAI*, vol. 4682, pp. 1120–1130, 2007.

[7] T. X. Lin and H. H. Chang, "Medical Image Registration Based on an Improved Ant Colony Optimization Algorithm," *Int. J. Pharma Med. Biol. Sci.*, vol. 5, no. 1, 2016.

[8] K. Krish, S. Heinrich, W. E. Snyder, H. Cakir, and S. Khorram, "A New Feature Based Image Registration Algorithm," *ASPRS 2008 Annu. Conf.*, 2008.

[9] Y.-W. Chen and A. Mimori, "Hybrid Particle Swarm Optimization for Medical Image Registration," *2009 Fifth Int. Conf. Nat. Comput.*, 2009.

Capítulo 2 Registración y fusión de imágenes

> Todas las verdades son fáciles de entender, una vez descubiertas. El caso es descubrirlas.
>
> *Galileo Galilei*

2.1 Introducción

El procesamiento digital de imágenes es una disciplina que se dividió del procesamiento digital de señales PDS, con la finalidad de dedicarse directamente al análisis e interpretación de la información proveniente de dispositivos de adquisición de imágenes y video. La primera prueba en la transmisión de una imagen se dio por los años 20, cuando se envió desde París a los EEUU una imagen. Sin embargo, los avances más significativos se dieron por medio de la NASA cuando recibía la digitalización de imágenes lunares enviadas por sondas y posteriormente por las misiones espaciales, siendo los primeros objetivos la reducción de ruido y corrección de distorsiones [1].

El resultado del procesamiento, manipulación, transformación e interpretación de una o varias imágenes siempre es otra imagen. La registración de imágenes no es la finalidad última dentro del PDI, más bien es un paso preliminar, pero de gran importancia pues permite que los procedimientos posteriores puedan alcanzar mejores resultados. Estos procedimientos pueden ser la segmentación, la reconstrucción, la morfología, el emparejamiento y la fusión de imágenes.

Dentro de las imágenes médicas se tiene como objetivo el análisis de la información, tanto a nivel local como global con la finalidad de mejorar el diagnóstico y seguimiento de patologías. Este análisis puede involucrar una o varias de las modalidades como son las ya mencionadas IRM y TAC además de las imágenes fMRI, SPECT y PET junto con las imágenes de ecografías, endoscopías, cámaras oculares, entre otras [2].

Además, la registración se utiliza para asistir en el seguimiento de enfermedades (efectividad de un tratamiento, crecimiento de tumores, etc.), comparación de información anatómica intra-paciente e inter-pacientes y finalmente, la generación de atlas anatómicos [3].

2.2 Imagen digital

Una imagen digital es una representación bidimensional o tridimensional discreta de una imagen, representada computacionalmente a partir de una matriz numérica generalmente en codificación binaria [1].

Una imagen natural depende de la cantidad de luminancia que existe en el ambiente. La luz, cuando incide en los fotorreceptores de un dispositivo de adquisición, genera un nivel de voltaje que permite su conversión a una imagen digital. En la **Fig. 2.1**, se muestra la representación de una imagen digital.

La matriz numérica I tiene un tamaño $m \times n$, donde m representa el número de filas y n el número de columnas. Una posición específica se representa como $I(i,j)$ y se denomina píxel[2], donde $i = 1 \dots m$ y $j = 1 \dots n$.

La definición formal de una imagen es: $I : E \rightarrow L$, donde I es la imagen, E es el dominio siendo un subconjunto finito no vacío perteneciente a los Z^2 y L el rango donde $L = \{l_{min}, \dots, l_{max}\}$ representando los niveles de intensidad de la imagen bajo la condición $l = |L|$. Una imagen en escala de grises de 8 bits tendría la siguiente representación $I : E \rightarrow \{0, \dots, 255\}$.

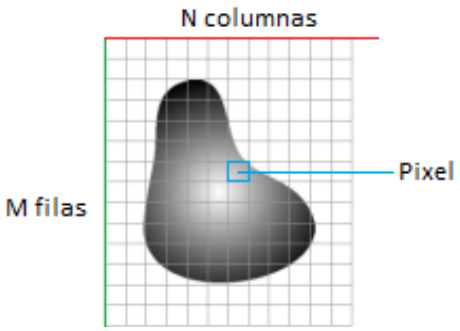


Fig. 2.1: Representación computacional de una imagen digital.

Las imágenes de color son imágenes multicanal representadas de manera general como $M : E \rightarrow \{L_1 \times L_2 \times \dots \times L_k\}$ siendo $M = (I_1, I_2, \dots, I_k)$ el conjunto de imágenes definido como $I_i \epsilon\, L_i^E$ donde $i = 1, \dots, k$. Para estas imágenes el valor de un píxel es una coordenada en el domino E que forma el vector $(I_1(t), I_2(t), \dots, I_k(t))$. Una imagen color RGB está formada por $k = 3$, donde los canales representan los colores primarios rojo, verde y azul. Formalmente está representada como $I_{RGB} = (I_1, I_2, I_3)$ y definida como $I_{RGB} : E \rightarrow \{L_1 \times L_2 \times L_3\}$, donde E sigue siendo un subconjunto no vacío perteneciente a los Z^2 y el rango para cada canal de 8 bits sería $L_1 = L_2 = L_3 = \{0, \dots, 255\}$.

2.3 Imágenes médicas

En el caso de las imágenes médicas los efectos físicos y aspectos técnicos para su adquisición difieren en gran medida al de las imágenes naturales, debido a que depende de la interacción de la radiación con la materia [2]. Este efecto físico permite medir la energía absorbida o dispersada durante la exposición de un tejido. Por ejemplo, en el caso de los Rayos X se analiza la atenuación diferencial (KeV) existente en los diferentes tejidos corporales. En medicina nuclear, en cambio, se analiza específicamente la absorción y dispersión de fotones, ya sea en exposición directa o con la ayuda de radiofármacos suministrados al paciente. En la **Fig. 2.2** se presentan algunas de las modalidades utilizadas en ciencias médicas.

En una imagen médica se pueden identificar básicamente tres tejidos biológicos: Óseo (hueso), tejido blando (agua) y tejido pulmonar (aire). Estas imágenes son principalmente de dos tipos: *anatómicas y funcionales* [3].

- **Las imágenes anatómicas** brindan información de la estructura, forma y masa de los tejidos. Dentro de este grupo se encuentran los rayos X, las imágenes RM y TAC, el ultrasonido (US), las secuencias de imágenes obtenidas en laparoscopía, laringoscopia y endoscopía.

[2] Píxel proviene de "picture elements" y es la mínima unidad de una imagen digital representada en un mapa de bits.

- **Las imágenes funcionales** brindan información de la actividad metabólica y funcional de órganos, tejidos y masas tumorales. Aquí se encuentran las imágenes SPECT, las PET y fMRI.

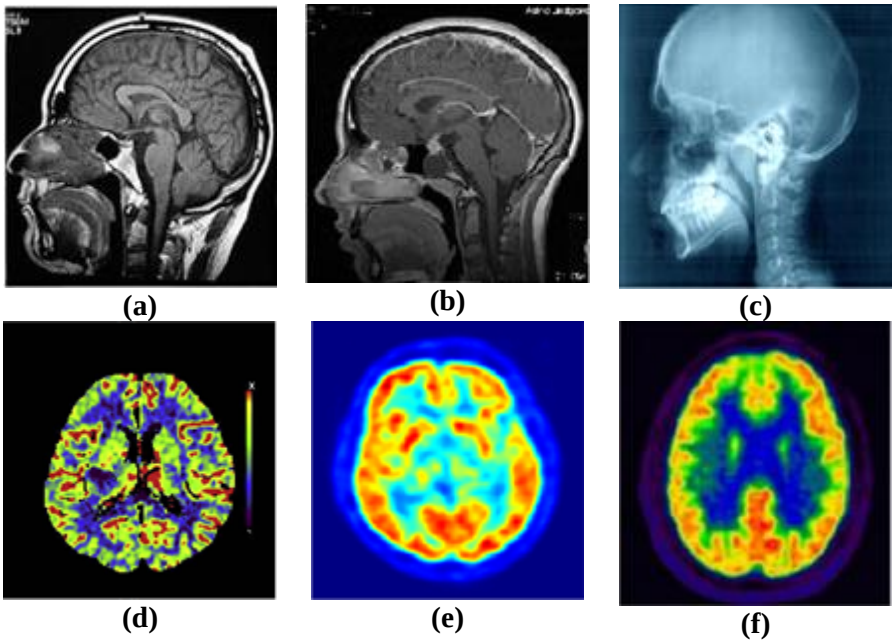


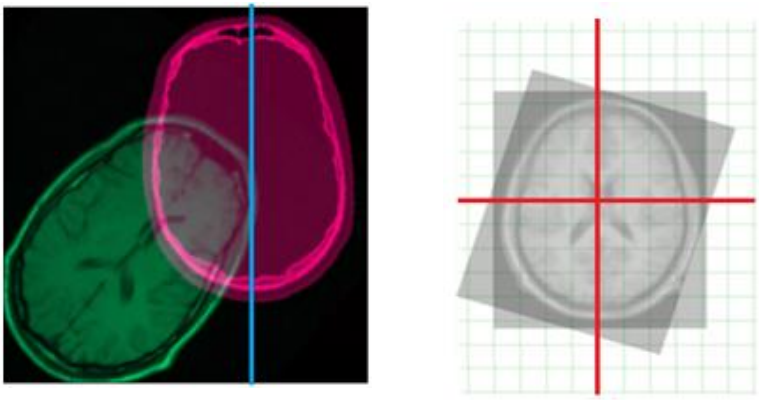
Fig. 2.2: Modalidades en imágenes médicas. **(a)** MRI **(b)** TAC **(c)** Rayos X **(d)** SPECT **(e)** PET **(f)** fMRI.

2.4 Registración de imágenes

La registración de imágenes es el proceso de alineamiento geométrico entre dos o más imágenes captadas en distintos instantes de tiempo, distintos puntos de vista o con distintos sensores con la finalidad de superponer la información visual relacionada entre ellas [4]. Una vez lograda la alineación se puede combinar, fusionar o completar la información de una misma escena. Este procedimiento tiene algunas áreas de aplicación, entre las más destacadas están la robótica, la teledetección y las ciencias médicas [5].

- En visión robótica, se la utiliza para análisis de ambientes y evasión de obstáculos, reconstrucción 3D de entornos, cálculo de profundidad de los objetos captados. Se puede aplicar también en sistemas de seguridad, para identificación de posibles intrusos.
- En teledetección, se utiliza para análisis de imágenes satelitales, con aplicaciones en la predicción de clima, en el análisis de fenómenos ambientales aplicados en la agricultura, efectos de cambios medioambientales o desarrollo urbanístico.
- En las ciencias médicas, el objetivo de esta tesis, se utiliza para analizar la información anatómica y funcional de tejidos. Este análisis puede ser aplicado en el diagnóstico, seguimiento y comparación de patologías. Así mismo, para la combinación y fusión de información con el objeto de automatizar y obtener exactitud en la comparación de información dentro de la misma modalidad (intra) como entre modalidades (inter).

En la **Fig. 2.3**, se muestra un ejemplo de registración de imágenes médicas de cerebro, lo que permite introducir varios de los elementos que intervienen en el proceso, lo cual es descrito en la siguiente sección.



(a) **(b)**

Fig. 2.3: Registración de imágenes médicas intermodalidad. **(a)** Imágenes de RM y TAC desalineadas. **(b)** Transformación geométrica realizada. **Fuente:** https://en.wikipedia.org/wiki/Image_registration.

2.4.1 Elementos de la registración

Los elementos que intervienen dentro del proceso son: las imágenes de entrada, el tipo de registración, el tipo de transformación geométrica, la medida de similaridad, el algoritmo de optimización y el interpolador. Maintz et al. [3] y Zitová et al. [6] describen con mayor detalle cada uno los elementos que intervienen en el proceso de registración.

1. **Las imágenes de entrada** son de dos tipos: la primera denominada *imagen de referencia* o base y la segunda la *imagen móvil* o deformada. La imagen de referencia permanecerá invariante durante el proceso, mientras que la móvil será transformada en sus coordenadas geométricas o espaciales hasta obtener la correspondencia entre todos los puntos y/o la información respecto de la imagen base.
2. **El tipo de registración** también está dividido en dos clases: *la monomodal*, es decir, entre imágenes de la misma modalidad i.e. MR-MR o CT-CT y *la multimodal*, es decir, entre imágenes de distinta modalidad i.e. MR-CT.
3. **El tipo de transformación geométrica** se dividen en las *transformaciones proyectivas lineales* y las *transformaciones elásticas* conocidas también como deformables. Estas transformaciones pueden aplicarse a nivel local o global en las imágenes.
   Las transformaciones proyectivas lineales cumplen con características matemáticas y geométricas propias, incluyendo al grupo de transformaciones **afines** que consisten en matrices con la última fila $(0,0,1)$ y las **Euclidianas** que contienen matrices 2×2 ortogonales [7]. En las transformaciones lineales se encuentran:
   a) **Transformación Rígida:** llamada también isometría en el plano $\mathbb{R}^2$, tiene 3 grados de libertad, donde permanece invariante la distancia euclidiana, el ángulo entre líneas y el área. La transformación afecta en la posición del objeto y su orientación.
   b) **Transformación de Similaridad:** es una isometría añadido un escalamiento de tipo isotrópico (i.e. escalado en la misma proporción en los ejes del plano $\mathbb{R}^2$), tiene 4 grados de libertad, donde permanece invariante la forma, el ángulo entre líneas, líneas paralelas.
   c) **Transformación Afín:** es una transformación lineal no singular seguida de una traslación. Tiene 6 grados de libertad correspondiente a los términos de la matriz de transformación la cual está compuesta por rotaciones y escalamiento no isotrópico. No preserva las líneas paralelas, pero sí la proporción de las distancias y la proporción de las áreas.

   d) **Transformación Proyectiva:** es la transformación lineal general no singular en coordenadas homogéneas por tanto tiene 8 grados de libertad. Preserva la relación cruzada de puntos colineales, pero se pierde la proporción de distancias y las líneas paralelas.

4. **La medida de similaridad** permite cuantificar la exactitud de la registración entre la imagen de referencia y la imagen registrada. La elección de esta medida depende directamente del tipo de registración, por ejemplo, para una registración monomodal se puede utilizar el coeficiente de correlación de Pearson [4], [8], mientras que, para el caso de una multimodal, se puede utilizar la medida de información mutua [4], [9].
5. **El algoritmo de optimización** es el núcleo de la registración y puede ser de dos tipos *determinista* y *estocástico*. De manera general, en los modelos de optimización determinista se asume que se conoce toda la información previamente para la resolución de un problema sin contemplar el azar dentro de la toma de decisiones. La programación lineal es un ejemplo de este tipo de optimización [10].
   Por el contrario, en los modelos estocásticos, conocidos también como probabilísticos, al menos una de las variables involucradas en la resolución del problema es tomada como un dato al azar. Este tipo de modelos permite reducir la cantidad de operaciones que realiza una computadora bajo un modelo determinístico, lo cual en muchos casos mejora el rendimiento a nivel general y la solución a un problema planteado. La programación evolutiva y la inteligencia de enjambres son ejemplos de optimización estocástica [11].
6. **El interpolador** dentro del PDI se utiliza cuando existe un cambio de tamaño o un re-mapeo de sus píxeles en una nueva grilla. El cambio de tamaño permite aumentar o disminuir el número de píxeles, mientras que un re-mapeo permite eliminar distorsiones, aplicar transformaciones espaciales o cambiar la perspectiva. Los interpoladores más utilizados son el de vecinos más cercanos, el bilineal y el bicúbico [12].

2.4.2 Definición formal de registración

Como se mencionó anteriormente, la registración es un problema de optimización. Dependiendo del punto de vista, se lo puede analizar como un problema de maximización o de minimización, donde se pretende calcular la mejor transformación geométrica, que permita maximizar una medida de similaridad para reducir la diferencia o error entre dos imágenes.

Matemáticamente está definida de la siguiente manera:

Sean 2 imágenes: $A : E \rightarrow L$ y $B : E \rightarrow L$, donde A es la imagen de referencia y B la móvil, E es el dominio $\Longrightarrow X \subset E : X \in \mathbb{Z}^2 \wedge X \neq \phi$ y L el rango siendo $L = \{0,255\}$

$$\begin{aligned} &\textit{Maximizar} && \mathcal{D}(B) \\ &\textit{Sujeto a} && \boldsymbol{\mathcal{T}}(A,B) : \mathbb{R}^2 \longrightarrow \mathbb{R}^2 \; o \; \mathbb{R}^3 \longrightarrow \mathbb{R}^3 \\ & && \mathcal{S}(B) \\ & && \mathcal{J}(B) = \mathcal{D}(B) + \alpha \, \mathcal{S}(B) \end{aligned}$$

donde $\mathcal{D}(B)$ es la medida de similaridad, $\boldsymbol{\mathcal{T}}(A,B)$ es la transformación geométrica entre las imágenes, $\mathcal{S}(B)$ es el factor de regularización y $\mathcal{J}(B)$ es el método de optimización.

2.5 Transformaciones geométricas

La geometría es una rama de las matemáticas que se enfoca en el estudio de un espacio (conjunto de puntos) junto con un grupo de transformaciones que son funciones del espacio en sí mismo y las estructuras que no varían en dicho grupo.

Dentro del procesamiento de imágenes, las transformaciones geométricas pueden ser vistas como funciones en el espacio ($\mathbb{R}^2$, $\mathbb{R}^3$) que redefinen la relación espacial entre los puntos de una imagen con el fin de manipular sus características y parámetros asociados (i.e. forma, tamaño). Tienen principal aplicación cuando se abordan temas como compensación de distorsión, registración de imágenes, normalización geométrica y mapeo de textura.

Básicamente, una transformación geométrica mapea un sistema de coordenadas para llevarlo a un nuevo sistema. En imágenes digitales, el mapeo se realiza por medio de las transformaciones espaciales, las cuales permiten obtener la correspondencia entre los puntos de la imagen de entrada respecto de la imagen de salida [3], [7].

2.5.1 El plano proyectivo 2D

Un punto en el plano se puede representar por un par de coordenadas $(x, y) \in \mathbb{R}^2$, de igual manera puede ser representado como un vector $[x\ y]$ considerando a $\mathbb{R}^2$ un espacio vectorial. Las entidades en geometría siempre serán representadas por vectores columna, por tanto, un punto se representa como $X = [x\ y\ w]^T$, donde w es parte de la notación de coordenadas homogéneas, la cual se utiliza para normalizar el sistema de coordenadas y establecer una escala de referencia. Esto se obtiene de la siguiente deducción:

Un punto $\boldsymbol{x} = (x, y)^T$ es parte de una línea $l = (a, b, c)^T$ si y solo si $ax + by + c = 0$, entonces representando por producto punto $(x, y, 1)(a, b, c)^T = 0$, esta constante 1 introducida para efectuar el producto puede ser descrita de manera general $(wx, wy, w)l = 0$ si y solo si $(x, y, 1)l = 0$, entonces el punto $\boldsymbol{x} = (x, y, w)^T$ en coordenada homogéneas pertenece al plano proyectivo $\mathbb{P}^2$, mientras que en el plano seria $\boldsymbol{x} = ({}^{x}/_{w}, {}^{y}/_{w})^T$ perteneciente a $\mathbb{R}^2$ [7], [13].

2.5.2 Transformaciones en el plano $\mathbb{P}^2$

Una transformación proyectiva conocida también como Homografía, forma un grupo de transformaciones llamado *grupo lineal proyectivo*, dentro de este grupo se encuentran las transformaciones afines y euclidianas [13], [14]. El orden de las transformaciones se detalla a continuación:

2.5.2.1 Isometría

Esta transformación preserva la distancia Euclídea en el plano $\mathbb{R}^2$, la cual tiene 3 grados de libertad, ya que se realizan cambios en 2 valores de traslación y 1 de rotación. Está representada de la siguiente manera en la **Ec. 2.1**:

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} \varepsilon\, cos\theta & -sin\theta & t_x \\ \varepsilon\, sin\theta & cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{2.1}$$

donde $\varepsilon = \pm 1$, θ el ángulo de rotación y $(t_x, t_y)^T$ el vector de traslación. Si $\varepsilon = 1$ la isometría mantiene la orientación y se la conoce también como transformación euclídea, si $\varepsilon = -1$ la isometría revierte la orientación y se la conoce como reflexión.

Esta transformación generalmente se realiza con objetos rígidos, es decir, con objetos que no sufren ningún tipo de deformación en su estructura. Las invariancias están en la longitud de las estructuras, el área y el ángulo entre líneas.

2.5.2.2 Similaridad

Esta transformación es una isometría, pero que adicionalmente realiza un escalado isotrópico. También se la conoce como *equiforme*, ya que conserva la forma de los objetos transformados y tiene 4 grados de libertad. Está representada de la siguiente forma en la **Ec. 2.2**:

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} s\,cos\theta & -s\,sin\theta & t_x \\ s\,sin\theta & s\,cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{2.2}$$

donde s es el valor del escalado a realizar, al ser isotrópico es el mismo valor en los dos ejes. En este caso lo que permanece invariante es la relación de las longitudes de las estructuras, los ángulos entre líneas y la relación de las áreas.

2.5.2.3 Afinidad

La afinidad es una transformación lineal no singular seguida de una traslación. Tiene 6 grados de libertad y puede ser calculada a partir de 3 puntos con sus correspondencias. Está representada de la siguiente manera en la **Ec. 2.3**:

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{2.3}$$

donde $\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$ es conocida como A que es una matriz no singular de 2×2, y está compuesta por rotaciones y escalados anisotrópicos, definida de manera general como lo muestra la **Ec. 2.4**:

$$A = R(\theta)\, R(-\phi) D\, R(\phi) \tag{2.4}$$

siendo $R(\theta)\, y\, R(\phi)$ rotaciones en los respectivos ángulos y D es una matriz diagonal para el escalado: $D = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$

En este tipo de transformaciones, por tener un escalado anisotrópico, no se preserva los ángulos y las longitudes. Las invariancias se dan en relación con las líneas paralelas, ya que, luego de la transformación, las líneas paralelas se intersecan en el infinito, por tanto, se puede decir que las líneas paralelas siguen siendo paralelas. Otra invariancia se da en la relación de longitud de las líneas paralelas y la relación de áreas, puesto que las rotaciones y los valores de escala no los afectan directamente.

2.5.2.4 Proyectiva

Es una transformación lineal no singular general, ya que generaliza una transformación afín. Tiene 8 grados de libertad y está representada de la siguiente manera:

$$x' = \begin{bmatrix} A & t \\ \mathrm{V}^T & v \end{bmatrix} x$$

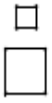
donde A es la matriz de transformación lineal no singular, igual que en el caso anterior, el vector t es el de traslación, $\mathrm{V} = (v_1,\ v_2)^T$ y v el factor de referencia al plano, generalmente $v = 1$. Entonces para calcular una transformación proyectiva se requiere 4 puntos con sus correspondencias, con 3 de ellos no colineales. Esta matriz está definida de acuerdo a la **Ec. 2.5**:

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ v_1 & v_2 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{2.5}$$

En esta transformación se mantiene invariante la relación cruzada de 4 puntos colineales, es decir, la relación cruzada de la longitud de una línea, también se mantiene las inflexiones y la concurrencia.

En la **Tabla 2.1**, se muestra el resumen de las transformaciones geométricas descritas, junto con una representación gráfica que permita visualizar el posible resultado al aplicar alguna de ellas.

Tabla 2.1: Resumen de las transformaciones geométricas en el plano.

Transformación	**Matriz**	**Distorsión** 	**Propiedades invariantes**
Isometría 2dof	$\begin{bmatrix} \varepsilon\, cos\theta & -sin\theta & t_x \\ \varepsilon\, sin\theta & cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix}$		Longitud, áreas.
Similaridad 3dof	$\begin{bmatrix} s\, cos\theta & -s\, sin\theta & t_x \\ s\, sin\theta & s\, cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix}$		Ángulos, relación de longitudes.
Afinidad 6dof	$\begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$		Paralelismo, relación de áreas, relación de longitud en puntos colineales.
Proyectiva 8dof	$\begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ v_1 & v_2 & 1 \end{bmatrix}$		Concurrencia, intersecciones, inflexiones y relación cruzada de longitud.

2.6 Medidas de Similaridad

Una medida de similaridad es una métrica que produce un valor alto a medida que se incrementa la dependencia entre los valores correspondientes de las secuencias analizadas [4]. Para ser considerada una métrica $\mathcal{D}$ tiene que cumplir con las siguientes propiedades:

- **Rango limitado**. $\mathcal{D}(x,y) \leq \mathcal{D}_0$, donde $\mathcal{D}_0$ es un número arbitrariamente grande y es el máximo valor que se puede obtener al comparar dos secuencias.
- **Reflexividad**. $\mathcal{D}(x,y) = \mathcal{D}_0$, si y solo si $x = y$.
- **Simetría**. $\mathcal{D}(x,y) = \mathcal{D}(y,x)$
- **Desigualdad triangular.** $\mathcal{D}(x,y)\mathcal{D}(y,z) \leq [\mathcal{D}(x,y) + \mathcal{D}(y,z)]\mathcal{D}(x,z)$

2.6.1 Coeficiente de correlación de Pearson

El coeficiente de correlación de Pearson (**Ec. 2.6**) es una medida estadística que mide la relación lineal entre pares de datos. Se denota con el símbolo r y está en el rango $-1 \leq r \leq 1$. Valores positivos de r se vinculan a una relación lineal positiva; los negativos, con relación lineal negativa; y el valor 0 cuando no existe relación lineal entre los datos. Mientras más cerca está r de 1 o -1, más fuerte es la relación de linealidad existente.

$$r = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{(x_i - \bar{x})(y_i - \bar{y})}{\sigma_x \sigma_y}\right) \tag{2.6}$$

El coeficiente de correlación es la relación entre la covarianza y las desviaciones estándar de las variables. Si x, y son intensidades de dos imágenes obtenidas de la misma escena con distintas condiciones lumínicas, la correspondencia de intensidades será lineal. Esta medida de similaridad es muy apropiada para comparación de imágenes del mismo tipo porque siempre estarán relacionadas linealmente, además de ser una medida muy eficiente, ya que se requieren mínimas sumas y multiplicaciones por cada píxel analizado [4], [15].

2.6.2 Correlación de rango de Spearman

Conocido también como Rho de Spearman (**Ec. 2.7**), es una métrica que permite calcular la relación de información entre dos variables mediante la definición de rangos. En el caso de las imágenes no se tienen porciones ordenadas y con saltos, sino los valores con distribución normal con el rango definido para una imagen color o escala de grises, por tanto, los valores de las intensidades reemplazan al ranking de datos. La fórmula para calcular es:

$$\rho = 1 - \frac{6\sum_{i=1}^{n}[R(x_i) - R(y_i)]^2}{n(n^2 - 1)} \tag{2.7}$$

Este coeficiente es poco sensible a puntos aislados, oclusiones y cambios de luminosidad, por lo que tiene buen rendimiento en el caso de calcular la correlación entre dos imágenes. Incluso por sus características permite aplicarlo con variables relacionadas no linealmente y en reconocimiento facial; sin embargo, el costo computacional que requiere es mayor que r la correlación de Pearson [4].

2.6.3 Información mutua

Es una medida de información basada en la teoría de las comunicaciones. Tiene como base a la entropía, la cual considera a un mensaje como una cadena de símbolos con s diferentes probabilidades de aparición por cada símbolo [16]. La entropía para una imagen está definida de acuerdo a la **Ec. 2.8**:

$$H = -\sum_{i} p_i \, log_2 \, p_i \tag{2.8}$$

La información mutua fue introducida por Shannon y utiliza el concepto de distribución de probabilidad conjunta PDF y de la entropía conjunta [17]. La ecuación para calcular la información mutua entre dos distribuciones uniformes está definida en la **Ec. 2.9***:*

$$S_{MI} = \sum_{i=0}^{255}\sum_{j=0}^{255} p_{ij} log_2 p_{ij}$$

$$S_{MI} = H_i + H_j - H_{ij} \tag{2.9}$$

La información mutua es muy utilizada en aplicaciones de registración multi modalidad, ya que calcula la dependencia de información no lineal como característica principal, alineando el histograma conjunto [17].

2.6.4 Relación de correlación (*correlation ratio*)

Es una medida de similaridad, propuesta por Pearson, que cuantifica el grado en el cual una variable Y es función de un solo valor de una variable X. Una propiedad importante de esta medida es que cuando decrementa la varianza de las intensidades en Y respecto las intensidades de X, la relación de correlación se incrementa. Esto lo hace útil en la comparación de imágenes que están relacionadas tanto lineal como no linealmente [7], [18]. Se calcula de la siguiente manera:

- Se recorren los valores de intensidades $i = 0 \ldots 255$ en X y se encuentran los valores de intensidades en las mismas posiciones en Y se lo denomina $Y[X_i]$.
- Se encuentra el valor promedio de Y, $m_i = \frac{1}{n_i} \sum_{X_i} Y[X_i]$.
- Se calcula la varianza de las intensidades de Y, $\sigma^2 = \frac{1}{n_i}\sum_{X_i}(Y[X_i] - m_i)^2$.
- Finalmente, se obtiene la relación de correlación, $\eta_{XY} = \sqrt{1 - \frac{1}{n}\sum_{i=0}^{255} n_i \sigma_i^2}$.

Esta medida de similaridad permite comparar imágenes captadas en distinta modalidad [18].

2.7 Interpolación y re-muestreo en imágenes

Interpolar significa partir de un conjunto de datos conocidos para estimar valores de datos no conocidos dentro del mismo conjunto. La interpolación en imágenes digitales es utilizada cuando se trata de un cambio de tamaño o un re-mapeo de sus píxeles en una nueva grilla. Un cambio de tamaño ocurre cuando se aumenta o disminuye el número de píxeles de la imagen, mientras que se realiza un re-mapeo para eliminar distorsiones, aplicar transformaciones espaciales y/o cambio de perspectivas.

En aplicaciones biomédicas, sin duda, la interpolación tiene un rol importante, por ejemplo, cuando se habla de modificar la frecuencia (tasa) de muestreo de los píxeles o vóxeles. A este procedimiento se le denomina *reescalado* y es muy común usarlo cuando los sistemas de adquisición tienen una resolución no homogénea por imagen. Otra aplicación se encuentra en el campo de la graficación por computadora cuando se trata de las texturas que componen un objeto renderizado [19].

En el caso específico de la registración de imágenes, ya que se realizan transformaciones geométricas a las imágenes (rotaciones, traslaciones, afines o proyectivas), la interpolación juega un rol importante previo obtener una registración de buena calidad. De igual manera, dentro del renderizado de imágenes médicas, debido a las operaciones geométricas involucradas, cuando es necesario volver a muestrear (*resampling*) el mapa de datos con las muestras obtenidas. Algunos procedimientos como la detección de bordes o los optimizadores, que se basan en la derivación de datos, también utilizan la interpolación para que sus operadores sean consistentes [2], [12].

Una interpolación lineal está definida mediante la **Ec. 2.10**:

$$f(x) = \sum_{k \in \mathbb{Z}^n} f_k \ \varphi_{int}(x-k) \qquad \forall\, \boldsymbol{x} = (x_1, x_2, \dots, x_n) \in \mathbb{R}^n \tag{2.10}$$

donde $f(x)$ es el valor interpolado de una coordenada x en un espacio de dimensión n, expresado por la combinación lineal de las muestras f_k evaluadas en las coordenadas enteras $\boldsymbol{k} = (k_1, k_2, \dots, k_n) \in \mathbb{Z}^n$ con los pesos dados por los valores de la función de síntesis $\varphi_{int}(x-k)$, donde φ_{int} puede ser una función $sinc$, $B-spline$, $Lineal$, $Lagrange$, entre otras.

Mientras que para realizar un remuestreo se parte de un grupo de datos discretos f_k para construir una interpolación mediante una función continua $f(x)$. Se realiza la transformación geométrica $\boldsymbol{T}\big(f(x)\big) = \sum f_{k_1} \ \varphi(T(x) - k_1)$, donde la función continua $f(\boldsymbol{T}(x))$ se resume en un conjunto de datos discretos $f(\boldsymbol{T}(k_2))$.

Los algoritmos de interpolación están divididos en dos grupos: adaptativos y no adaptativos. Estos algoritmos se utilizan para interpolar una imagen sin perder el contenido visual de la misma. Los algoritmos adaptativos interpolan en función a características de la imagen, por ejemplo, bordes, textura. Son algoritmos adaptativos *Qimage, Genuine Fractals*. Los no adaptativos, en cambio, interpolan por el promedio de sus píxeles vecinos, dándole la misma relevancia a cada uno de ellos. Son algoritmos no adaptativos: vecino más cercano, lineal, bilineal, bicúbico, spline, sinc [20].

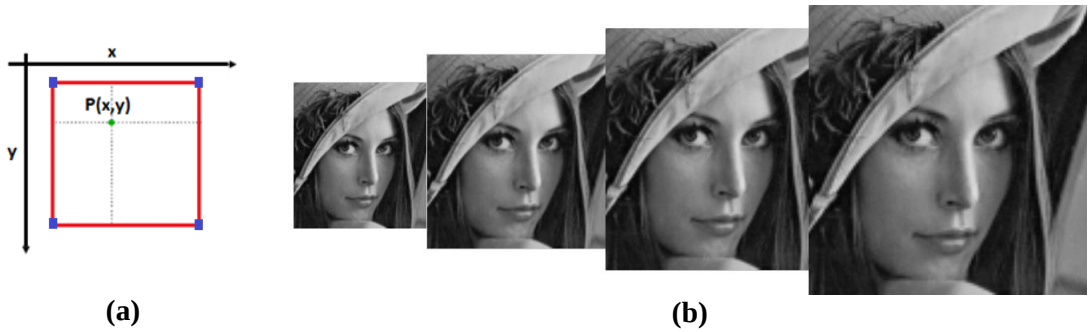
2.7.1 Interpolación por vecino más cercano

Este es el procedimiento más básico de interpolación, en el que simplemente se considera al píxel más cercano al punto de interpolación como salida. Esto hace que sea más rápido y computacionalmente eficiente. Sin embargo, muchas veces los resultados no son óptimos, ya que los píxeles se amplían durante el proceso. En la **Fig. 2.4**, se muestra un ejemplo de cambio de escala utilizando esta interpolación.

Fig. 2.4: Imagen de Lena deformada en el espacio de escala usando la interpolación de vecinos cercanos.

2.7.2 Interpolación bilineal

En esta interpolación se considera a los 4 píxeles más cercanos de forma cuadrangular, es decir, píxeles de la forma 2×2, entonces se promedia dichos píxeles para obtener el píxel interpolado. En la **Fig. 2.5**, se muestra la metodología para el cálculo junto con un ejemplo de cambio de escala utilizando esta interpolación.



(a) **(b)**

Fig. 2.5: Interpolación bilineal. **(a)** Representación geométrica para el cálculo del nuevo píxel. **(b)** Imagen de Lena deformada en el espacio de escala usando interpolación bilineal.

2.7.3 Interpolación bicúbica

Esta interpolación utiliza los 16 píxeles más cercanos de forma cuadrangular, con una distribución de la forma 4×4, los píxeles más cercanos tienen un mayor peso en el cálculo del píxel interpolado. Este método permite obtener imágenes más nítidas de salida, respecto de los anteriores, además que optimiza el tiempo de cómputo. En la **Fig. 2.6**, se muestra la metodología junto con un ejemplo de cambio de escala utilizando esta interpolación.

(a) **(b)**

Fig. 2.6: Interpolación bicúbica. **(a)** Representación geométrica para el cálculo del nuevo píxel. **(b)** Imagen de Lena deformada en el espacio de escala usando interpolación bicúbica.

2.8 Métodos generales de registración

Existen distintos enfoques y métodos con los que se ha abordado la temática de la registración de imágenes y que sirven como base para el desarrollo de esta investigación. Los métodos tienen como base la matemática y especialmente la geometría tanto en 2D como 3D, sin olvidar las técnicas de PDI desarrolladas con profundidad en los últimos años [2], [3].

2.8.1 Basada en puntos

Este método está basado en conocer previamente un conjunto de puntos y sus correspondencias entre las dos imágenes[3], para luego, mediante la registración, buscar su alineamiento. Cuando estos puntos son confiables y tienen como propósito la registración se los conoce como "fiduciario" o "puntos fiduciarios" [21].

Los fiduciarios pueden ser puntos relevantes de la anatomía conocidos como marcadores anatómicos, puesto que son distinguibles y únicos en el contenido de la imagen como el surco central con la línea media del cerebro o la intersección de una estructura lineal con una superficie. Por ejemplo, la unión de septos en un seno paranasal. Algunas aplicaciones para este método podrían ser la neurocirugía craneal o el posicionamiento del tornillo pedicular.

2.8.2 Basada en marcadores

Este método tiene la ventaja de que los puntos fiduciarios no dependen de las características anatómicas relevantes, sino que pertenecen a objetos externos ubicados en posiciones específicas y conocidas durante la adquisición de la imagen, específicamente para acelerar el proceso de registración. Igual que en el caso anterior, lo que se pretende es la alineación de dichos puntos. Los objetos usados son marcadores de cráneo o piel, ubicados en el paciente, siendo de un material adecuado para ser distinguible en la imagen por su forma o número atómico [22].

2.8.3 Basada en características invariantes

Este método busca automáticamente características distintivas en la imagen como puede ser bordes, esquinas, contornos, puntos de intersección entre otros con la finalidad de describir características propias junto a su entorno. La ventaja es que se analizan las características que no varíen expuestas al cambio de escala, orientación y traslación de la imagen. En la bibliografía actual se pueden encontrar muchos extractores de características siendo los más representativos SIFT [23], [24] y SURF [25]. Una vez encontradas las correspondencias entre las características de las imágenes se procede a calcular la función de transformación.

2.8.4 Basada en superficies

Este método está basado en la caracterización de los contornos anatómicos, los cuales no son complejos de extraer. Una superficie sería un conjunto de puntos con información en común a los que se calculan sus propiedades geométricas. Especialmente utilizada en registración 3D, lo que se establece es una correspondencia de superficies para el cálculo de la función de transformación. Por ejemplo, la superficie del contorno de la piel es usada en registración multimodal CT-MR, la superficie del rostro y huesos es igualmente muy utilizada debido a su relativa fácil y automática extracción en modalidades como la RM y TAC [2], [26].

[3] Un punto esta descrito en coordenadas (x, y) para una imagen bidimensional o (x, y, z) para una tridimensional, siendo $x, y, z \in R$ y corresponden al tamaño de la imagen.

2.8.5 Basada en intensidad

Este método está basado directamente en los valores escalares de los píxeles o vóxeles de la imagen, los cuales dependen directamente de la modalidad de adquisición. Basado en la teoría de la información se definen medidas de similaridad para comparar la cantidad de información correlacionada entre las imágenes. Entre ellas están la información mutua usada en registración multimodal [27], la correlación cruzada [28], la diferencia de intensidad de imágenes, entre otras. Entonces el proceso de registración se lleva a cabo mediante la maximización o minimización de una de estas medidas de similaridad. Por tanto, el proceso es iterativo pretendiendo encontrar el valor óptimo de la transformación.

2.9 Fusión de imágenes

La fusión de imágenes es un área de investigación independiente que ha captado igualmente gran interés en la actualidad debido a la variedad de sensores existentes para la adquisición de imágenes. Su principal objetivo es la combinación de información proveniente de imágenes que han sido captadas en distinta modalidad, distintos instantes de tiempo y bajo distinta perspectiva. Las aplicaciones están enfocadas a la fotografía, vigilancia, investigación médica, entre otras [29].

El proceso de fusión implica combinar información de dos o más imágenes de distinta modalidad en una sola imagen de alta calidad que contenga la máxima cantidad de información sin producir detalles inexistentes o información adicional a las imágenes de entrada [30]. Para alcanzar este objetivo, se deben tener en cuenta los siguientes enunciados:

- La imagen fusionada debe conservar la información más relevante y complementaria de las imágenes de entrada.
- Se debe evitar que el proceso de registración previo sea ineficiente, es decir, que la diferencia entre la imagen de referencia y la móvil sea muy grande.
- De acuerdo con la aplicación es necesario contar con un buen filtrado de ruido.

En la **Fig. 2.7**, se muestra a manera de ejemplo, la fusión de imágenes naturales en el espectro visible e infrarrojo.

(a) **(b)** **(c)**

Fig. 2.7: Fusión de imágenes. **(a)** Imagen en el espectro visible. **(b)** Imagen de la misma escena captada en el espectro infrarrojo. **(c)** Imagen resultante o fusionada. **Fuente:** J. Ma et al., Infrared and visible image fusion methods and applications: A survey, Information Fusion, 45, pp. 153-178, 2019.

2.9.1 Enfoques para Fusión de imágenes

Basándose en la literatura, la fusión puede ser abordada desde dos enfoques distintos: uno basado *en el dominio del espacio* y el otro basado *en el dominio de la transformación*. En ambos casos, se espera que la imagen fusionada sea más apropiada y entendible tanto para los humanos como para los procesos de aprendizaje computacional.

En el dominio espacial, se trabaja directamente con la información de los píxeles de las imágenes de entrada y se los combina de manera lineal o no lineal. Matemáticamente este proceso se muestra en la **Ec. 2.11**.

$$I_F = \emptyset(I_1, I_2, \dots, I_N) = \alpha_1 I_1 + \alpha_2 I_2 + \cdots + \alpha_N I_N \tag{2.11}$$

donde I_F es la imagen fusionada, $I_1, I_2, \dots, I_N$ las N imágenes de entrada, $\emptyset$ la regla aplicada para la fusión, α es una constante tal que $\sum_{i=1}^{N} \alpha_i = 1$ [29].

Para el caso del dominio de la transformación, las imágenes de entrada son transformadas del dominio espacial a otro más adecuado aplicando la transformación más adecuada, por ejemplo, transformada de Fourier, wavelets o pirámides. Matemáticamente se puede expresar mediante la **Ec. 2.12.**

$$I_F = T^{-1}(\emptyset(T(I_1), T(I_2), \dots, T(I_N))) \tag{2.12}$$

donde I_F es la imagen fusionada, T^{-1} es el operador inverso de la transformación T.

La transformación inversa se utiliza para reconstruir la imagen original. Por ejemplo, si se utiliza la transformada de Fourier denominada T en el proceso de fusión, la obtención de la imagen resultante o fusionada estará basada en la transformada inversa de Fourier T^{-1} [29], [30].

2.9.2 Niveles de abstracción para Fusión de imágenes

Otra manera de abordar el proceso de fusión de imágenes es a través de los distintos niveles de abstracción. Los algoritmos desarrollados cumplen 3 niveles distintos: el de los píxeles, el de las características (*features*) y el de decisión.

2.9.2.1 Nivel de píxeles

Este es uno de los niveles usados ampliamente para abordar esta temática. En este nivel, la fusión se realiza directamente en la capa de datos original, por tanto, la cantidad de información se conserva. Para esto se requiere una alta precisión en el proceso de registración previo a este nivel, lo cual incrementa la probabilidad de obtener una buena fusión [31].

En este nivel se encuentran los siguientes métodos: *de promedios ponderados*, basados en *análisis de componentes principales* (PCA), en lo referente al dominio del espacio [32], [33]. Referentes al dominio de transformación están: *basados en wavelets* como las Haar, Daubechies, Mexican, Hat, Morlet, y Meyer [34], [35]; y los basados en *pirámides Laplacianas* [36].

2.9.2.2 Nivel de características

En este nivel de abstracción se procesan los objetos existentes dentro de las imágenes. Por tanto, se considera a las características locales o globales de la imagen y los objetos más importantes que la información de los píxeles individuales.

La fusión se realiza con base en las regiones de la imagen [29]. Este procedimiento es más robusto que los enfoques basados en píxeles, para adaptarse a las reglas de fusión inteligente [37], [38].

2.9.2.3 Nivel de decisión

El nivel de decisión es el más alto dentro del proceso de fusión de imágenes. Este nivel comprende la combinación de información una vez que cada sensor ha procesado y realizado una

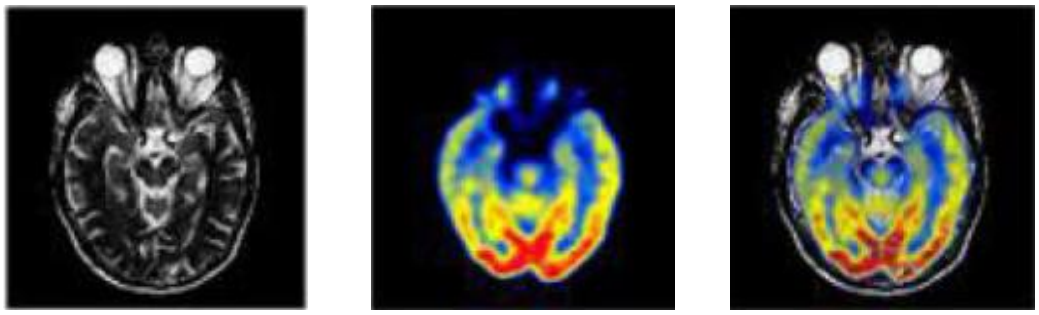
determinación preliminar (ubicación, atributos, identidad entre otros) en las imágenes de entrada. La fusión en este nivel utiliza las salidas de la detección y clasificación de objetos iniciales como entradas al algoritmo de fusión [39].

Este nivel tiene distintas aplicaciones: en teledetección e imágenes remotas [40], [41], así como en aplicaciones industriales de multi sensores [42], [43].

2.9.3 Fusión de imágenes médicas

Dentro del campo médico, la fusión de imágenes es un área de investigación con un gran crecimiento en los últimos años debido a su amplio rango de aplicaciones. La fusión de imágenes médicas permite combinar múltiples imágenes de una o varias modalidades para obtener una imagen de alta calidad que reduzca la aleatoriedad y redundancia en la información. Este proceso se ha aplicado en el diagnóstico y evaluación de problemas médicos, donde se ha logrado un incremento en la precisión clínica para la toma de decisiones basadas en imágenes médicas [44].

Existen varias modalidades de imágenes médicas que se utilizan como entradas para el proceso de fusión (**Sección 2.3**). La modalidad utilizada depende específicamente de la información del órgano motivo del estudio clínico. En la **Fig. 2.8**, se muestra la fusión de imágenes en distinta modalidad.



(a) **(b)** **(c)**

Fig. 2.8: Fusión de imágenes médicas. **(a)** Imagen MRI de cerebro en secuencia T2. **(b)** Imagen PET en el mismo corte que la Fig. (a). **(c)** Imagen resultante o fusionada.

Dado que resulta casi imposible captar todos los detalles desde una modalidad de imagen que garantice precisión y solidez en el análisis y diagnóstico requerido, el enfoque que se utiliza es la observación de varias modalidades para evaluar los detalles que se complementan entre sí. Esto permite que el experto médico pueda realizar una evaluación más confiable y precisa [44], [45].

2.9.4 Métodos de Fusión de imágenes médicas

La fusión, como se mencionó anteriormente, consta de la etapa de alineamiento o registración de dos o más imágenes de entrada y, posteriormente, la combinación de su información. Varias investigaciones descritas en la bibliografía han permitido el desarrollo de los métodos presentados a continuación.

2.9.4.1 Métodos morfológicos

Este es uno de los métodos más explorados para la fusión de imágenes médicas y es usado para detectar la información espacial relevante de las imágenes de entrada. Está basado principalmente en la estructura de los operadores morfológicos que definen las operaciones de apertura y cierre, aunque en ciertas aplicaciones se pueden usar los operadores de morfología de torres y pirámides, transformadas K-L, entre otros [46].

Los métodos morfológicos se han aplicado en imágenes de cerebro con modalidades en TAC y RM [47], [48].

2.9.4.2 Métodos basados en wavelets

Estos métodos utilizan las wavelets para extraer la información relevante de una imagen e insertarla en otra. La información relevante está dada en términos de alta frecuencia, donde las wavelets son apropiadas, ya que tienen la característica principal de extraer las frecuencias tanto en el dominio del espacio como del tiempo. Por tanto, la imagen fusionada contiene las mejores características (*features*) de las imágenes de entrada [44].

Este método tiene varias aplicaciones. Entre ellas está el diagnóstico médico [49], [50], además de la combinación con otras técnicas como el *Análisis de componentes individuales* (ICA) [51].

2.9.4.3 Métodos basados en conocimiento

En imágenes médicas, hay varios casos en los que el conocimiento del médico puede utilizarse para diseñar algoritmos de segmentación y registración de imágenes. Estos métodos depositan la confianza en el experto médico para el etiquetado e identificación del conocimiento relevante para la tarea de fusión [52].

En la mayoría de los casos estos sistemas se comparan con un modelo estándar para confirmar los resultados obtenidos. Estos métodos se pueden utilizar en combinación con otros como la intensidad de píxeles [32], [33] y los sistemas embebidos [53].

2.9.4.4 Métodos basados en lógica difusa

La conjunción y disyunción son propiedades de la lógica difusa que han sido ampliamente utilizadas dentro del procesamiento de imágenes [44]. También ha sido útil para la fusión de imágenes, aplicándose como un operador de transformación de características [29] o un operador de decisión [39] para el procedimiento de fusión.

La lógica difusa también puede dar mayor soporte a los métodos basados en conocimiento y adaptarse eficientemente con las redes neuronales. Las aplicaciones están dadas en fusión multimodal [54], [55], detección de tumores [56] y recuperación de imágenes médicas [57].

2.9.4.5 Métodos basados en redes neuronales

Las redes neuronales tienen la capacidad de predecir, analizar e inferir información. Luego de un proceso de entrenamiento, un dato puede ser determinado sin requerir una expresión matemática explícita. Esto hace que la red neuronal sea atractiva para la fusión de imágenes, ya que la naturaleza de la variabilidad entre las imágenes está sujeta a cambios cada vez que se utiliza una nueva modalidad [44].

Aunque las redes neuronales ofrecen generalidad, su robustez está limitada por la calidad de los datos y la precisión de la convergencia del algoritmo durante el entrenamiento. Para mejorar esto se emplean métodos híbridos de redes neuronales y procesamiento secuencial con otras técnicas de fusión, tales como, la lógica difusa, los algoritmos genéticos, máquinas de soporte vectorial (SVM), entre otros.

Las aplicaciones de fusión de imágenes bajo este método son muy amplias: para sistemas multi sensor [58], diagnóstico médico [59] y fusión multimodal [60].

Referencias

[1] R. C. Gonzalez and R. E. Woods, "Digital Image Processing," 2008.

[2] J. M. Fitzpatrick, D. L. G. Hill, and C. R. Maurer, "Image Registration," in *Handbook of Medical Imaging, Volume2: Medical Image Processing and Analysis*, 2000, pp. 447–514.

[3] J. B. A. Maintz and M. A. Viergever, "A survey of medical image registration," *Med. Image Anal.*, vol. 2, no. 1, pp. 1–36, 1998.

[4] A. Ardeshir Goshtasby, *Image Registration: Principles, Tools and Methods (Advances in Computer Vision and Pattern Recognition)*, Springer. 2012.

[5] R. Szeliski, "Computer Vision: Algorithms and Applications," 2010.

[6] B. Zitová and J. Flusser, "Image registration methods: a survey," *Image Vis. Comput.*, vol. 21, pp. 977–1000, 2003.

[7] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. 2003.

[8] A. Nakhmani and A. Tannenbaum, "A new distance measure based on generalized Image Normalized Cross-Correlation for robust video tracking and image recognition," 2013.

[9] J. P. W. Pluim, J. B. A. Maintz, and M. A. Viergever, "Mutual information based registration of medical images: a survey," *IEEE Trans. Med. Imaging*, 2003.

[10] D. Luenberger and Y. Ye, *Linear and Nonlinear Programming*, Third. 2008.

[11] A. P. Engelbrecht, "Computational intelligence: An introduction," in *Studies in Computational Intelligence*, 2nd ed., 2007.

[12] P. Thévenaz, T. Blu, and M. Unser, "Image Interpolation and Resampling," *Handb. Med. Imaging, Process. Anal.*, pp. 393–420, 2000.

[13] C. C. Remsing, "Transformation Geometry," Grahamstown, South Africa, 2006.

[14] G. Wolberg, "Geometric Transformation Techniques For Digital Images: A Survey," New York, 1988.

[15] A. Nakhmani and A. Tannenbaum, "A new distance measure based on generalized Image Normalized Cross-Correlation for robust video tracking and image recognition," 2013.

[16] J. P. W. Pluim, J. B. A. Maintz, and M. A. Viergever, "Mutual information based registration of medical images: a survey," *IEEE Trans. Med. Imaging*, 2003.

[17] R. S. Suganya R., Priyadharsini K., "Intensity Based Image Registration by Maximization of Mutual Information," *Int. J. Comput. Apl.*, vol. 1, no. 20, 2010.

[18] A. Roche, G. Malandain, X. Pennec, and N. Ayache, "The Correlation Ratio as a New Similarity Measure for Multimodal Image Registration," *LNCS*, vol. 1496, pp. 1115–1124.

[19] F. M. Weinhaus and V. Devarajan, "Texture Mapping 3D Models of Real-World Scenes," *ACM Comput. Surv.*, vol. 29, no. 4, pp. 325–365, 1997.

[20] C. J. Moses, D. Selvathi, and V. M. A. Sophia, "VLSI Architectures for Image Interpolation: A Survey," *VLSI Des.*, vol. 2014, pp. 1–10, 2014.

[21] J. B. West, J. M. Fitzpatrick, S. A. Toms, C. R. Maurer, and R. J. Maciunas, "Fiducial Point Placement and the Accuracy of Point-based, Rigid Body Registration," *Neurosurgery*, vol. 48, pp. 810–817, 2001.

[22] M. V Wyawahare, P. M. Patil, and H. K. Abhyankar, "Image Registration Techniques: An overview," *Int. J. Signal Process. Image Process. Pattern Recognit.*, vol. 2, no. 3, 2009.

[23] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[24] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, 1999, pp. 1150–1157 vol.2.

[25] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-Up Robust Features (SURF)," *Comput. Vis. Image Underst.*, vol. 110, no. 3, pp. 346–359, 2008.

[26] L. Gottesfeld Brown, "A Survey of Image Registration Techniques," *ACM Comput. Surv.*, vol. 24, pp. 325–376, 1992.

[27] Q. Li and I. Sato, "Multimodality Image Registration by Particle Swarm Optimization of Mutual Information," *LNAI*, vol. 4682, pp. 1120–1130, 2007.

[28] L. Z. L. Zheng, Y. W. Y. Wang, and C. H. C. Hao, "Cross-correlation registration algorithm based on the image rotation and projection," in *4th International Conference on Image and Signal Proccesing*, 2011.

[29] B. Meher, S. Agrawal, R. Panda, and A. Abraham, "A survey on region based image fusion methods," *Inf. Fusion*, vol. 48, pp. 119–132, Aug. 2019

[30] D. K. Sahu and M. P. Parsai, "Different Image Fusion Techniques - A Critical Review," *Int. J. Mod. Eng. Res.*, vol. 2, no. 5, pp. 4298–4301, 2012.

[31] B. Yang, Z. Jing, and H. Zhao, "Review of pixel-level image fusion," *J. Shanghai Jiaotong Univ.*, vol. 15, no. 1, pp. 6–12, Feb. 2010.

[32] M. Masthanaiah, P. Janardhan, and S. Kumar, "Development of Image Fusion Algorithms by Integrating PCA, Wavelet and Curvelet Transforms," *Int. J. Innov. Res. Comput. Commun. Eng. (An ISO*, vol. 3297, no. 4, 2007.

[33] R. Kaur, M. T. Student, F. Sahib, and S. Kaur, "An Approach for Image Fusion using PCA and Genetic Algorithm," *Int. J. Comput. Appl.*, vol. 145, no. 6, p. 54, 2016.

[34] G. Pajares, J. Us, and M. De La Cruz, "A wavelet-based image fusion tutorial," *Pattern Recognit.*, vol. 37, pp. 1855–1872, 2004.

[35] V. Sahu and D. Sahu, "Image Fusion using Wavelet Transform: A Review," *Glob. J. Comput. Sci. Technol.*, vol. 14, no. 5, pp. 21–28, 2014.

[36] T. Rama Murthy and V. P. S Naidu, "Image Fusion for Enhanced Vision System using Laplacian Pyramid," *Int. J. Eng. Res. Technol.*, vol. 4, pp. 507–512, 2015.

[37] D. Egfin Nirmala and V. Vaidehi, "Comparison of Pixel-level and feature level image fusion methods," *2nd Int. Conf. Comput. Sustain. Glob. Dev.*, 2015.

[38] H. Li, L. Li, and J. Zhang, "Multi-focus image fusion based on sparse feature matrix decomposition and morphological filtering," *Opt. Commun.*, vol. 342, pp. 1–11, May 2015.

[39] N. B. Kolekar and R. P. Shelkikar, "Decision level based Image Fusion using Wavelet Transform and Support Vector Machine," *Int. J. Sci. Eng. Res.* , vol. 4, pp. 54–58, 2015.

[40] M. Fauvel, J. Chanussot, and J. A. Benediktsson, "Decision Fusion for the Classification of Urban Remote Sensing Images," *IEEE Trans. Geosci. Remote Sens.*, vol. 44, no. 10, pp. 2828–2838, Oct. 2006.

[41] B. Luo, M. M. Khan, T. Bienvenu, J. Chanussot, and L. Zhang, "Decision-Based Fusion for Pansharpening of Remote Sensing Images," *IEEE Geosci. Remote Sens. Lett.*, vol. 10, no. 1, 2013.

[42] R. C. Luo, S. H. H. Phang, and K. L. Su, "Multilevel multisensor based decision fusion for intelligent animal robot," in *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No.01CH37164)*, vol. 4, pp. 4226–4231.

[43] R. Heideklang and P. Shokouhi, "Decision-Level Fusion of Spatially Scattered Multi-Modal Data for Nondestructive Inspection of Surface Defects.," *Sensors (Basel).*, vol. 16, no. 1, Jan. 2016.

[44] A. P. James and B. V Dasarathy, "Medical Image Fusion: A survey of the state of the art," *Inf. Fusion*, 2014.

[45] B. V. Dasarathy and B. V., "Information fusion in the realm of medical applications – A bibliographic glimpse at its growing appeal," *Inf. Fusion*, vol. 13, no. 1, pp. 1–9, Jan. 2012.

[46] N. Uniyal and S. K. Verma, "Image Fusion using Morphological Pyramid Consistency Method," *Int. J. Comput. Appl.*, vol. 95, no. 25, pp. 34–38, Jun. 2014.

[47] G. K. Matsopoulos and S. Marshall, "Application of Morphological Pyramids: Fusion of MR and CT Phantoms," *J. Vis. Commun. Image Represent.*, vol. 6, no. 2, pp. 196–207, Jun. 1995.

[48] G. K. Matsopoulos, "Multiresolution morphological fusion of MR and CT images of the human brain," *IEE Proc. - Vision, Image, Signal Process.*, vol. 141, no. 3, p. 137, 1994.

[49] J. Srikanth and C. N. Sujatha, "Image Fusion Based On Wavelet Transform For Medical Diagnosis," *J. Eng. Res. Appl.*, vol. 3, no. 6, pp. 252–256, 2013.

[50] S. Chavan, A. Pawar, and S. Talbar, "Multimodality Medical Image Fusion using Rotated Wavelet Transform," in *Proceedings of the International Conference on Communication and Signal Processing 2016 (ICCASP 2016)*, 2017.

[51] Z. Cui, G. Zhang, and J. Wu, "Medical Image Fusion Based on Wavelet Transform and Independent Component Analysis," in *2009 International Joint Conference on Artificial Intelligence*, 2009, pp. 480–483.

[52] A. Galande and R. Patil, "The art of medical image fusion: A survey," in *2013 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2013, pp. 400–405.

[53] S. Thoma, A. Rettinger, and F. Both, "Knowledge Fusion via Embeddings from Text, Knowledge Graphs, and Images," Apr. 2017.

[54] Chung-Hsien Huang and Jiann-Der Lee, "Improving MMI with enhanced-FCM for the fusion of brain MR and SPECT images," in *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, 2004, pp. 562-565 Vol.3.

[55] A. Villéger, L. Ouchchane, J.-J. Lemaire, and J.-Y. Boire, "Data Fusion and Fuzzy Spatial Relationships for Locating Deep Brain Stimulation Targets in Magnetic Resonance Images," Springer, Berlin, Heidelberg, 2006, pp. 909–919.

[56] W. Dou, S. Ruan, Q. Liao, D. Bloyet, J.-M. Constans, and Y. Chen, "Fuzzy Information Fusion Scheme Used to Segment Brain Tumor from MR Images," Springer, Berlin, Heidelberg, 2006, pp. 208–215.

[57] X. Tai and W. Song, "An Improved Approach Based on FCM Using Feature Fusion for Medical Image Retrieval," in *Fourth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2007)*, 2007, pp. 336–342.

[58] A. Starzacher and B. Rinner, "Embedded Realtime Feature Fusion based on ANN, SVM and NBC," *12th Int. Conf. Inf. Fusion*, 2009.

[59] Q. P. Zhang, W. J. Tang, L. l. Lai, W. C. Sun, and K. P. Wong, "Medical diagnostic image data fusion based on wavelet transformation and self-organising features mapping neural networks," in *Proceedings of 2004 International Conference on Machine Learning and Cybernetics (IEEE Cat. No.04EX826)*, pp. 2708–2712.

[60] Yang-Ping Wang, Jian-Wu Dang, Qiang Li, and Sha Li, "Multimodal medical image fusion using fuzzy radial basis function neural networks," in *2007 International Conference on Wavelet Analysis and Pattern Recognition*, 2007, pp. 778–782.

Capítulo 3 Optimización

> Necesitamos especialmente de la imaginación en las ciencias. No todo es matemáticas y no todo es simple lógica, también se trata de un poco de belleza y poesía.
>
> *María Montessori*

3.1 Introducción

La optimización está presente en cualquier tipo de problema que involucre una toma de decisión. Esto permite elegir una alternativa a partir de un conjunto de opciones posibles. Esta elección está basada en el deseo de tomar *la mejor* decisión de acuerdo con algún criterio [1].

El área de la optimización ha tomado gran interés en los últimos años debido, entre otros aspectos, al avance tecnológico que ha permitido el desarrollo de tecnología computacional de alto rendimiento, tanto a nivel *hardware* con procesadores en paralelo de alta velocidad y *software* con las herramientas de aprendizaje computacional.

Los problemas de optimización pueden ser tan cotidianos como la elección de la ruta que requiera menor tiempo o esfuerzo para llegar a un destino hasta problemas complejos de ingeniería, como el diseño de un robot capaz de aprender de su entorno utilizando eficientemente sus recursos energéticos. Una amplia cantidad de aplicaciones tiene su fundamento en la optimización, entre estas, sin duda se encuentra el procesamiento de imágenes, el reconocimiento de patrones y el aprendizaje computacional.

3.2 Optimización matemática

Un problema de optimización involucra la selección de valores para un número de parámetros interrelacionados, enfocado en un solo objetivo diseñado para cuantificar el rendimiento y medir la calidad de la decisión. El objetivo será maximizar (o minimizar, dependiendo del planteo) una función matemática sujeta a algunas restricciones que pueden limitar la selección de los valores de decisión [2]. Formalmente está definido de la siguiente manera (**Ec. 3.1)**:

$$\begin{aligned} &\textit{Minimizar} && f_0(x) \\ &\textit{Sujeto a} && f_i(x) \leq b_i, \quad i = 1, 2, \dots, m. \end{aligned} \tag{3.1}$$

donde el vector $x = (x_1, x_2, \dots, x_n)$ es la *variable de optimización* del problema, la función $f_0: \mathcal{R}^n \to \mathcal{R}$ es la *función objetivo,* las funciones $f_i: \mathcal{R}^n \to \mathcal{R}$ son las *funciones de restricción* y las constantes b_i son los *límites* para las restricciones [3].

Un vector x^*es denominado el *óptimo* o la solución al problema de optimización, si tiene el valor más pequeño de la función objetivo entre todos los vectores resultado que satisfacen las restricciones. Matemáticamente se podría expresar que: para cualquier k con $f_1(k) \leq b_1, \dots, f_m(k) \leq b_m$, se tiene un $f_0(k) \geq f_0(x^*)$.

3.2.1 Aplicaciones

En el ajuste de datos, el objetivo es encontrar un modelo a partir de un conjunto de potenciales modelos que se adapte mejor a los datos observados y a la información previa. En este caso, las variables son los parámetros del modelo y las restricciones pueden representar la información previa o los límites para los parámetros (por ejemplo, valores no negativos). La función objetivo deberá ser una medida para predecir el error entre los datos observados y los datos predichos por el modelo.

A modo de ejemplo, podemos mencionar el caso del ajuste de curva polinomial presentado por Bishop en [4]. Se tiene un conjunto de N observaciones para la variable de optimización x, definidos de la siguiente manera: $\boldsymbol{x} = (x_1, \dots, x_N)^T$ y las correspondientes observaciones $\boldsymbol{t} = (t_1, \dots, t_N)^T$. En la **Fig. 3.1**, se muestran los valores dados para $N = 10$.

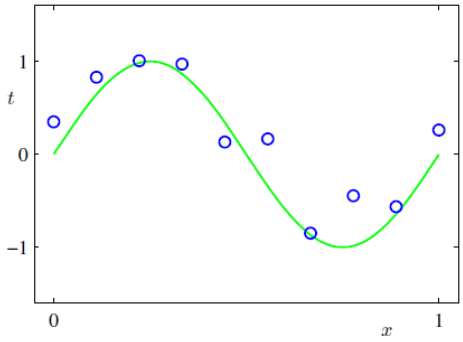


Fig. 3.1: La función matemática $y = sin\,(2\pi x)$, expresada en línea verde. El vector $\boldsymbol{x}$ corresponde a la observación de N valores mostrado en círculos azules, mientras que $\boldsymbol{t}$ son los datos objetivos. **Fuente:** C. Bishop, Pattern Recognition and Machine Learning. Cap. 1, pag. 4, 2006.

El objetivo es predecir $\boldsymbol{t}$ para nuevos valores de $\boldsymbol{x}$ a partir de esta información y sin un conocimiento previo de la función matemática. Por lo tanto, el modelo de ajuste polinomial se puede ajustar a partir de la **Ec. 3.2.**

$$y(x, \boldsymbol{w}) = w_0 + w_1 x + w_1 x^2 + \cdots + w_M x^M = \sum_{i=0}^{M} w_i x^i \tag{3.2}$$

Los valores de los coeficientes $\boldsymbol{w}$ serán calculados, con la finalidad de reducir el error entre los valores predichos por el modelo $y(x, \boldsymbol{w})$ y los valores objetivos o del grupo de entrenamiento $\boldsymbol{t}$. Esta medida de error se calcula con la **Ec. 3.3.**

$$E(\boldsymbol{w}) = 0.5 \sum_{n=1}^{N} \{y(x, \boldsymbol{w}) - t_n\}^2 \tag{3.3}$$

La elección del orden del polinomio M, para el cual los valores de $\boldsymbol{w}$ permitan obtener un valor de error lo más pequeño como sea posible.

En la **Fig. 3.2**, se muestran algunos modelos usados para ajustar el modelo presentado. Para los valores $M = 0$ y $M = 1$ se obtienen modelos polinomiales con un bajo ajuste de los datos. Con $M = 3$, el modelo se ajusta cercanamente a la función $\sin(2\pi x)$, mientras que con $M = 9$, el error es $E(\boldsymbol{w}) = 0$, pero el modelo obtenido es muy deficiente del esperado. Este modelo genera una sobre optimización durante el proceso.

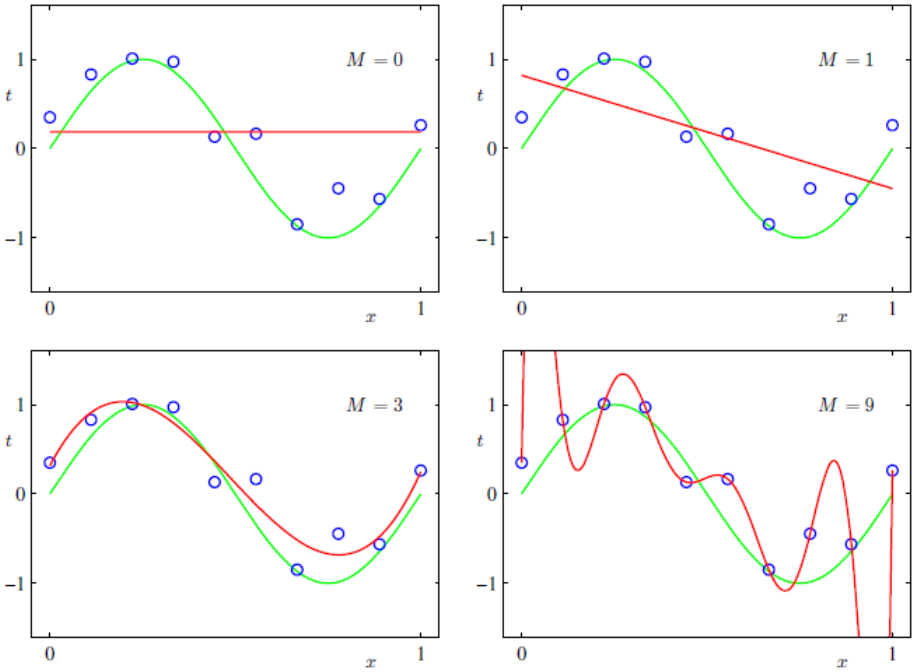
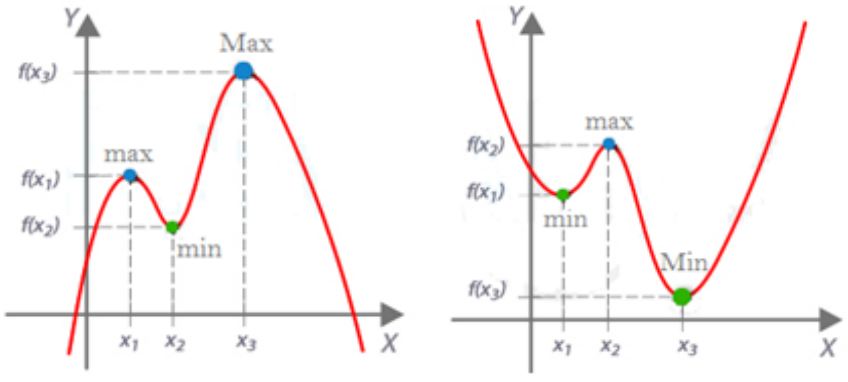


Fig. 3.2: Ajuste de datos por medio de polinomios de grado 0, 1, 3 y 9. A medida que el grado aumenta, el error disminuye; sin embargo, el caso de M=9, los datos no se ajustan al modelo, más bien generan una sobre optimización. Para el ejemplo M=3 es el mejor modelo para ajustar los datos. **Fuente:** C. Bishop, Pattern Recognition and Machine Learning. Cap. 1, pag. 7, 2006.

3.3 Extremos de una función

Los extremos de una función, conocidos también como máximos o mínimos, son los valores más grandes o pequeños que puede tomar una función en un determinado punto ya sea dentro de una determinada región (extremo local) o en el dominio total (extremo global). Localizar los valores extremos de una función es el objetivo de la optimización [28].

En la **Fig. 3.3**, se presentan dos funciones matemáticas en las que se identifican los extremos de cada una de ellas.



(a) **(b)**

Fig. 3.3: Extremos de una función. **(a)** Función a maximizar: Max es el máximo global. **(b)** Función a minimizar: Min es el mínimo global. En ambos casos max y min son extremos locales. **Fuente:** http://matemolinillo.blogspot.com/2018/05/maximos-y-minimos.html.

Un extremo local se define formalmente de la siguiente manera: Sea $f: A \subset \boldsymbol{\mathcal{R}}^n \rightarrow \boldsymbol{\mathcal{R}},\ x_0 \in A$ y $P = (x_0, f(x_0))$ donde P es un punto en la gráfica de la función. Se dice que P es un **máximo local** si existe un entorno reducido de centro x_0 denotado como $E'(x_0)$, donde para todo $x \in E'(x_0)$ se cumple que $f(x) > f(x_0)$ o un **mínimo local** si se cumple $f(x) < f(x_0)$.

Un extremo global se define formalmente de la siguiente manera: Sea $f: A \subset \boldsymbol{\mathcal{R}}^n \rightarrow \boldsymbol{\mathcal{R}},\ x_0 \in A$ y $P = (x_0, f(x_0))$ donde P es un punto en la gráfica de la función. Se dice que P es un **máximo global** si para todo $x \neq x_0, x \in A$ se cumple que $f(x) > f(x_0)$ o un **mínimo global** si se cumple $f(x) < f(x_0)$.

3.4 Métodos de optimización

En general, los métodos de optimización están enfocados a la solución de problemas que pueden ser de: programación lineal, restricciones, sin restricciones y programación no lineal.

3.4.1 Programación lineal

La programación lineal es uno de los campos que abarca la mayor cantidad de los problemas de optimización con cierta complejidad. Se centra en la maximización o minimización de una función lineal denominada *función objetivo*, cuando el sistema de ecuaciones e inecuaciones es lineal [2].

Debido a su simplicidad en las matemáticas, la riqueza en su teoría y la forma simple de calcular las soluciones, lo hacen muy popular y atractivo. La forma estándar de representar a la programación lineal se muestra en la **Ec. 3.4**.

$$\begin{aligned} &\textit{Minimizar} && c_1x_1 + c_2x_2 + \cdots + c_nx_n \\ &\textit{Sujeto a} && a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ & && a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ & && \quad \ldots \qquad \ldots \qquad\quad \ldots \qquad \ldots \\ & && a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_m \\ &\textit{Y} && x_1 \geq 0, x_2 \geq 0, \ldots, x_n \geq 0 \end{aligned} \tag{3.4}$$

donde b_i, c_i *y* a_{ij} son constantes reales y x_i son los valores reales a ser determinados en el proceso.

Uno de los métodos tradicionales para resolver este tipo de problemas es el método Simplex [5]. Dentro del procesamiento de imágenes, la programación lineal ha sido utilizada con varios propósitos, entre ellos está la segmentación y la reconstrucción de imágenes [6], [7].

3.4.2 Optimización con restricciones

Bajo este método, la maximización o minimización de la función objetivo con respecto a las variables de optimización tienen ciertas restricciones. Estas restricciones pueden ser *duras*, en cuyo caso establecen condiciones para que cumplan las variables o *suaves*, donde algunos valores para las variables se penalizan en la función objetivo [8].

En la práctica, generalmente los problemas son de tipo restrictivo. Esto debido a que un problema global puede ser dividido en subproblemas, cada uno de ellos tendrá un alcance limitado por las restricciones impuestas a las variables de optimización. Por ejemplo, en un problema de planificación, las restricciones presupuestarias se imponen comúnmente para separar ese problema de uno más global [4].

La formulación matemática para este método es presentada en la **Ec. 3.5.**

$$\begin{aligned} &\textit{Minimizar} && f(\boldsymbol{x}) \\ &\textit{Sujeto a} && h_i(\boldsymbol{x}) = 0, \quad i = 1, 2, .., m \\ & && g_j(\boldsymbol{x}) \leq 0, \quad j = 1, 2, \dots, r \\ & && \boldsymbol{x} \in S \end{aligned} \tag{3.5}$$

donde $\boldsymbol{x}$ es el vector n-dimensional de incógnitas, f, h_i y g_i son las funciones reales de las variables $x_1, x_2, \dots, x_n$ y S es un subconjunto del espacio n-dimensional. f es la función objetivo y h_i , g_i son el conjunto de restricciones.

Las aplicaciones dentro del procesamiento de imágenes están enfocadas a la mejora del contraste [9], registración no rígida [10] y reconstrucción de imágenes [11].

3.4.3 Optimización sin restricciones

La optimización sin restricciones surge directamente en algunos problemas específicos, pero también surgen indirectamente de las reformulaciones de los problemas de optimización con restricciones.

En cierto modo, se puede considerar que este tipo de optimización carece de una estructura matemática adecuada para su aplicación en casos reales; sin embargo, se ha demostrado todo lo contrario. Por ejemplo, si se amplía el alcance de un problema, las restricciones podrían desvanecerse. O, a su vez, porque un problema con restricciones fácilmente se puede convertir en uno sin ellas [4].

Un aspecto importante de la optimización continua (ya sea con o sin restricciones) es que las funciones son suaves, lo que significa que las segundas derivadas existen y son continuas. Matemáticamente está definida como lo muestra la **Ec. 3.6.**

Sea $x \in \boldsymbol{R}^n$ *un vector de valores reales con componentes* $n \geq 1$, *por tanto* $f: \boldsymbol{\mathcal{R}}^n \to \boldsymbol{\mathcal{R}}$ *es una función suave.*

$$min_x\, f(x) \tag{3.6}$$

De acuerdo con la complejidad de la función f, a menudo se utilizan métodos iterativos para explorar el espacio de soluciones de manera efectiva. En métodos iterativos, generalmente, el cálculo de una nueva solución está determinado por un paso *(step)* en una dirección de movimiento de la solución actual. En la **Ec. 3.7**, se muestra la ecuación general de estos métodos [12].

$$x_{i+1} = x_i + \alpha \boldsymbol{d} \tag{3.7}$$

donde x_{i+1} es la nueva solución potencial al problema de optimización, x_i es la solución actual, α es un valor real positivo y $\boldsymbol{d}$ es un vector de desplazamiento.

Cuando la dirección del desplazamiento está basada en el gradiente de la función, los métodos son llamados *por descenso de gradiente* [13]. A modo de ejemplo de optimización, se presenta la función de *Ackley* (**Fig. 3.4**), la cual se expresa matemáticamente para n-dimensiones según la **Ec. 3.8.** Esta función cumple con las características diferenciales requeridas y tiene una única solución global (unimodal).

$$f(x) = -20\exp\left(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d}x_i^2}\right) - \exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi x_i)\right) + 20 + \exp(1) \qquad (3.8)$$

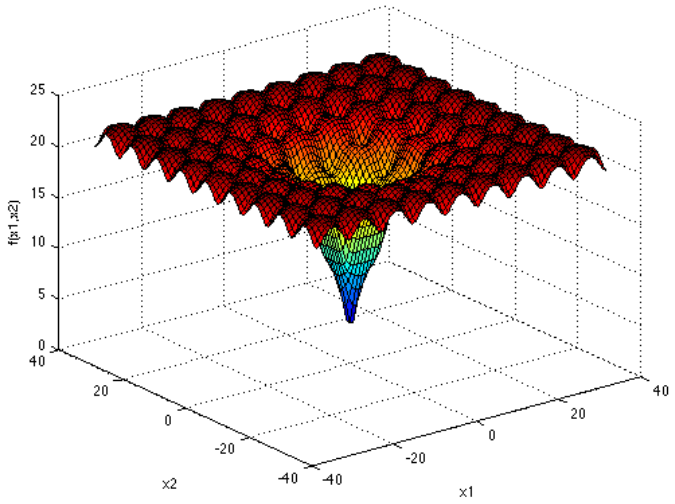


Fig. 3.4: Función Ackley en 2D utilizada para testear algoritmos de optimización.

En general los métodos iterativos tienen amplias aplicaciones, tanto en matemática como en ingeniería, economía, entre otras áreas. Entre las aplicaciones que se pueden realizar en procesamiento de imágenes, con el método de descenso de gradiente está la registración [14], la recuperación [15] y compresión [16] de imágenes.

3.4.4 Programación no lineal

La programación no lineal es muy similar a la lineal, pues los elementos que la componen son la función objetivo, las restricciones generales y las variables límite. La diferencia principal radica en que presenta al menos una función no lineal ya sea en la función objetivo o en las ecuaciones o inecuaciones de las restricciones [17].

Existen muchos sistemas reales que deben ser modelados mediante este método de optimización; sin embargo, esto es más complicado que con los métodos anteriores, entre otras razones, debido a que es más difícil distinguir entre un óptimo local y el global y podría haber regiones del espacio de búsqueda que no estén conectadas. De acuerdo con la inicialización se podría conducir a diferentes soluciones finales ya que es difícil satisfacer la igualdad de las restricciones.

Formalmente, un problema de optimización no lineal está definido en el **Ec. 3.9.**

$$\begin{aligned} &\textit{Minimizar} && f(\boldsymbol{x}) \\ &\textit{Sujeto a} && h_i(\boldsymbol{x}) = 0, \quad i = 1, 2, .., m \\ & && g_j(\boldsymbol{x}) \leq 0, \quad j = 1, 2, \dots, r \\ & && x \subseteq R^n \end{aligned} \tag{3.9}$$

donde $\boldsymbol{x}$ es el vector de incógnitas en n-dimensiones, f es la función objetivo y h_i , g_i constituyen el conjunto de restricciones.

Dentro del procesamiento de imágenes se utiliza para la restauración de imágenes [18], [19].

3.5 Modelos para optimización

Los problemas de optimización, en general, pueden ser abordados mediante dos tipos de enfoques: *el determinístico y el estocástico.* Cada uno de estos, utilizan los métodos de optimización descritos en la sección anterior.

- **Enfoque determinístico:** cuando todas las variables del problema pueden ser conocidas con certeza, lo que implica que no existe incertidumbre sobre los parámetros del modelo para la toma decisiones.
- **Enfoque estocástico:** cuando alguna de las variables del problema no puede ser conocida con anterioridad, lo cual implica incertidumbre en los parámetros del modelo. Por tanto, los modelos se conocen como *probabilísticos* y las variables como *estocásticas o aleatorias.*

Los modelos determinísticos siguen una rigurosidad matemática para que sus variables generen salidas en el contexto y rangos esperados. En general, están diseñados para pruebas de hipótesis o para crear sistemas de gestión para disminuir la incertidumbre. Sin embargo, su rendimiento y aplicabilidad puede ser notablemente disminuida cuando aumenta la complejidad del modelo junto con la cantidad de variables y la relación entre ellas. Los algoritmos más relevantes son: Simplex, Newton, quasi-Newton, de direcciones conjugadas y Levenberg-Marquardt [20], [21].

Los modelos estocásticos, por el contrario, contemplan que al menos una de sus variables sea de tipo aleatorio, por tanto, el azar y el principio de incertidumbre están involucrados. La probabilidad y estadística permiten modelar adecuadamente los procesos estocásticos que pueden ser utilizados en grandes series de muestreos optimizando tiempo y el uso de recursos computacionales [22], [23].

El presente trabajo aborda la registración de imágenes médicas mediante técnicas de optimización estocástica. Por tanto, el resto del capítulo abordará con más detalle los algoritmos de optimización basados en este enfoque. Entre los algoritmos estocásticos más conocidos se encuentran: la búsqueda aleatoria (RS), el recocido simulado (SA), la búsqueda tabú, la programación genética, la computación evolutiva y la inteligencia de enjambres [24].

3.5.1 Optimización estocástica

La optimización estocástica, en estadística, se refiere a la aplicación de métodos para minimización o maximización de una función objetivo cuando existe aleatoriedad. La aleatoriedad puede estar vinculada a la función de costo o el conjunto de restricciones [23].

La aleatoriedad tiene como ingrediente clave a la incertidumbre para la toma de decisiones. La incertidumbre se caracteriza generalmente por una distribución de probabilidad en los parámetros. Aunque esta pueda estar definida rigurosamente, en la práctica puede variar desde posibles resultados de los datos, hasta distribuciones de probabilidad conjuntas específicas y precisas. Además, cuando algunos de los datos son aleatorios, las soluciones y el valor óptimo para el problema de optimización son también aleatorios [25].

Una distribución de decisiones óptimas generalmente no es implementable. En algunos casos, puede ser más apropiado tratar de encontrar una decisión que garantice que un conjunto de restricciones se mantendrá con cierta probabilidad. Formalmente está definida como lo muestra la **Ec. 3.10.**

$$\begin{aligned} &\textit{Minimizar} && f(\boldsymbol{x}) \\ &\textit{Sujeto a} && h_i(\boldsymbol{x}) = 0, \quad i = 1, 2, .., m \\ & && \Pr\left[g_j(\boldsymbol{x}) \leq 0\right] \geq \alpha, \quad j = 1, 2, \dots, r \\ & && x \subseteq R^n \end{aligned} \tag{3.10}$$

donde $\boldsymbol{x}$ es el vector de incógnitas en n-dimensiones, f es la función objetivo y h_i , g_i constituyen el conjunto de restricciones. g_i está definida con base en una función de distribución de probabilidad y α es el valor de la probabilidad que está el rango $(0, 1]$.

La optimización estocástica ha sido utilizada en un amplio rango de aplicaciones, entre ellas: la planificación de la producción manufacturera y de la capacidad de generación eléctrica, gestión del tráfico, modelización y planificación macroeconómica.

3.5.2 Optimización heurística

El desarrollo de dispositivos computacionales con altas prestaciones ha permitido que los métodos estocásticos sean cada vez más robustos, pero también han dado lugar a la generación de una subrama de algoritmos estocásticos conocidos como *heurísticos*.

Formalmente se puede definir a la optimización heurística de la siguiente manera:

Dado un conjunto $\boldsymbol{A} \subset R^{\boldsymbol{n}}$ y una función continua $f: \boldsymbol{A} \to \mathcal{R}$ denota que $\boldsymbol{A}^* = argmin\, f = \{a \in \boldsymbol{A}: f(a) \leq f(x)\}$. Si $\boldsymbol{A}$ es compacto, entonces $\boldsymbol{A}^*$ es siempre no vacío.

En este tipo de optimización se pretende encontrar soluciones que aproximen a $\boldsymbol{A}^*$, mediante la aleatorización de soluciones candidatas en el espacio de soluciones del problema de optimización. Mediante un método computacional conocido como *metaheurística*, se ejecuta un algoritmo iterativo que mejora la solución candidata. Durante el proceso, la actualización de la mejor solución se basa en una regla que puede ser aprendida por la experiencia o establecida previamente, conocida como *heurística* [26].

Estos algoritmos tienen grandes aplicaciones dentro del aprendizaje de máquina, inteligencia computacional, reducciones de costo y las ciencias médicas.

3.5.2.1 Características principales

Estos algoritmos están inspirados en los conceptos y el funcionamiento de varios sistemas biológicos que se encuentran en la naturaleza, entre ellos, la genética, los procesos evolutivos o la inteligencia colectiva de algunos agentes como los peces o las hormigas.

Su fácil implementación y su versatilidad para adaptarse a diferentes clases de problemas los hacen cada vez más atractivos. Sus principales características son:

- **Disponibilidad:** en continuo crecimiento debido a los recursos computacionales de los que se dispone actualmente.
- **Determinísticos:** debido a su versatilidad pueden aplicarse en esta modalidad.
- **Eficiencia:** a pesar de que no se puede asegurar que se encontrará el óptimo global en todos los casos.
- **Calidad de la solución:** la solución entregada por estos algoritmos siempre será de alta calidad incluso cuando no se obtenga el óptimo global.
- **Multipropósito:** pueden ser utilizados en optimización unimodal donde existe una única solución, pero también en multimodal donde la optimización es multiobjetivo.

En general, el objetivo que persiguen es encontrar la mejor solución al problema de optimización, aunque no siempre se pueda asegurar la obtención de la solución exacta; sin embargo, una aproximación estocástica de alta calidad de un óptimo global es probablemente más valiosa que un mínimo local determinista de baja calidad proporcionado por un método clásico.

3.5.2.2 Metaheurística vs. heurística

La palabra heurística proviene del verbo griego *heuriskein* (ενρισκειν), el cual significa *encontrar o descubrir,* por tanto, tiene que ver con la regla con la cual se encuentran nuevas soluciones potenciales durante el proceso iterativo. Se establece mediante el análisis del funcionamiento del sistema biológico y su posterior planteo a nivel matemático y computacional. La heurística puede ser aprendida por la experiencia, es decir, mediante las mejores acciones o decisiones tomadas durante un determinado tiempo o establecidas inicialmente de acuerdo con el tipo de algoritmo.

Mientras que "Meta" significa *más allá* o en un nivel superior, por tanto, tiene que ver con las estrategias que se establecen para la búsqueda del optimo global. Este concepto permite un nivel de abstracción del problema de optimización. Finalmente, la metaheurística puede incorporar mecanismos para evadir o evitar zonas confinadas en el espacio de búsqueda.

Por ejemplo, en el caso del algoritmo inspirado en la colonia de hormigas ACO, la heurística se basa en la selección proporcional de acuerdo con la función de aptitud (*roulette-wheel selection*) y la metaheurística se basa en la matriz de feromonas que determina el funcionamiento colectivo de la colonia en búsqueda del óptimo global [27].

3.5.2.3 Método de búsqueda local

Los algoritmos heurísticos pueden dividirse en dos métodos principales, los cuales están basados de acuerdo al objetivo que persiguen. Estos son los *métodos de construcción* y los *métodos de búsqueda local.*

Los algoritmos en los que se enfoca este trabajo son los de búsqueda local. La característica principal de los algoritmos que pertenecen a este grupo es que exploran el espacio de soluciones

de manera no sistemática. La heurística de cada uno de ellos determina la forma en que se moverá a través del espacio de soluciones de la función.

De manera general, un algoritmo heurístico que maximiza una función objetivo está representado de la siguiente manera:

1. **Generar** una solución actual x^c
2. **Mientras** el criterio de finalización no sea alcanzado:
3. **Seleccionar** $x^n \in \mathcal{N}(x^c)$
4. **Si** $f(x^n) > f(x^c)$:
5. $x^c = x^n$
6. **Fin**

donde $\mathcal{N}$ es el conjunto de soluciones potenciales vecinas a x^c, $f(x^c)$ y $f(x^n)$ representan los valores de la función de costo para cada una de las soluciones potenciales. El criterio de finalización se refiere al número de iteraciones, el tiempo de ejecución o alguna medida estadística.

3.5.2.4 Exploración vs. Explotación

En los métodos heurísticos, la solución al problema se encuentra mediante la experiencia y el descubrimiento ya que una búsqueda exhaustiva, en muchos casos, es impráctica debido a la complejidad involucrada. Para la búsqueda de estas soluciones en el espacio de una función es necesario establecer un equilibrio entre la exploración y la explotación, el cual es crucial cuando se habla de algoritmos de optimización.

Se entiende por exploración como *diversificación*, es decir, la habilidad para visitar muchas y diferentes regiones del espacio y por explotación como *intensificación*, es decir, la habilidad de obtener soluciones de buena calidad en dichas regiones [29].

Los algoritmos genéticos basan su criterio de exploración mediante *cruce* y la explotación mediante *mutación.* En el algoritmo ACO [27], la exploración se basa en las *feromonas* y la explotación en *la información heurística.* En otras palabras, la exploración permite realizar la búsqueda a nivel global y la explotación a nivel local. Este criterio de equilibrio tiene también otras aplicaciones, por ejemplo, la computación evolutiva, el aprendizaje de máquina y el control óptimo [30], [31].

3.6 Inteligencia de enjambres

La inteligencia de enjambres se encarga del estudio de los sistemas biológicos que trabajan colectivamente como las colonias de hormigas, los cardúmenes de peces, las bandadas de aves, las colmenas de abejas entre otras. Mientras que la inteligencia computacional de enjambres se encarga de modelar matemáticamente dichos sistemas con el fin de aplicarlos en diversos problemas, especialmente en el área de la optimización estocástica y heurística.

Un *enjambre* se puede definir como un grupo homogéneo de agentes móviles que se comunican de manera directa o indirecta, para actuar sobre su entorno local [32]. De manera formal se puede decir que la inteligencia de enjambres es una propiedad de un sistema formado por agentes simples y limitados en su capacidad individual, pero que, al actuar con su entorno local permiten obtener patrones de comportamiento cooperativos globales al realizar tareas necesarias para su supervivencia. A este proceso se lo llama *emergente*, ya que de comportamientos individuales no

tan sofisticados emerge el comportamiento complejo global del enjambre. Es por ello por lo que también se la conoce como *inteligencia colectiva* [33].

Los enjambres trabajan de manera descentralizada y bajo pocas reglas. Descentralizada porque no tienen un agente para controlar o dirigir el comportamiento individual y colectivo. La autoorganización es el tema principal con restricciones limitadas a las interacciones entre ellos. Los agentes de un enjambre interactúan o se comunican entre sí para lograr un comportamiento colaborativo coherente.

La comunicación puede ser directa por medio del contacto visual o auditivo como por ejemplo la danza de las abejas; o puede ser indirecta, cuando un agente cambia el entorno y los demás responden a ese cambio, por ejemplo, un rastro dejado como guía en el entorno. Este tipo de comunicación se la conoce como *estigmergia*. La estigmergia está definida como la colaboración a través del medio físico. En sistemas tales como las colonias de hormigas, los agentes colaboran a través de pautas o hitos dejados: feromonas, acumulación de objetos o la temperatura [34].

3.6.1 Inteligencia computacional de enjambres

La inteligencia de enjambres ha permitido el avance de áreas como la inteligencia computacional y las ciencias de la computación, ya que sus conceptos han llegado a ser innovadores y de gran interés para diversas aplicaciones. SI está basada en propiedades y características específicas de la naturaleza y/o de las poblaciones, lo cual produce soluciones de bajo costo computacional, rápidas y robustas [35].

Varios de los enjambres que han servido de inspiración para los modelos de SI son:

- **Las hormigas:** por la asignación de tareas de manera dinámica sin ningún coordinador.
- **Las termitas:** por la capacidad de crear grandes estructuras de nidos que supera la comprensión de cada una de ellas.
- **Las aves y los peces:** por la autoorganización en patrones espaciales óptimos. En el caso de las aves, mediante la percepción visual y los sonidos, mientras que, en el caso de los peces se realiza mediante la dirección y la velocidad del movimiento.
- **Las abejas:** por la comunicación visual mediante danzas que resulta en el comportamiento óptimo para su alimentación.
- **Las bacterias:** por la comunicación mediante moléculas (comparables a las feromonas) para realizar un seguimiento colectivo de los cambios en su entorno.

El objetivo de los modelos computacionales de inteligencia de enjambres es modelar los comportamientos simples de los individuos y las interacciones locales con el entorno y los individuos vecinos, a fin de obtener comportamientos más complejos que puedan usarse para resolver problemas complejos, en su mayoría problemas de optimización [33].

La inteligencia de enjambres ha sido aplicada con éxito en diversos trabajos de investigación, entre los que se incluye la optimización de funciones, búsqueda de rutas óptimas, la programación y análisis de imágenes y datos [36], el aprendizaje automático [37], la bioinformática e informática médica [38], e incluso en las finanzas y negocios [39].

3.6.1.1 Principios generales

Para la generación la inteligencia computacional de enjambres hay que considerar los principios evolutivos de selección, expuestos en [40]:

- **Principio de proximidad:** el enjambre debe ser capaz de analizar de manera simple el entorno que lo rodea. Este análisis se traduce en cálculos simples del espacio y el tiempo que representan el costo energético que el enjambre deberá utilizar para llevar a cabo una tarea. Los cálculos se refieren a la respuesta conductual directa a los estímulos del entorno, los cuales, permiten maximizar la utilidad de todo el enjambre al realizar una determinada actividad. Estas actividades pueden variar de acuerdo con la complejidad de los agentes involucrados, pero algunas de ellas pueden ser: buscar y recuperar alimento, construir nidos, movimientos colectivos y defensa del grupo.
- **Principio de calidad:** el enjambre además de responder a las consideraciones del entorno que lo rodea, debe ser capaz de responder a factores de calidad como, por ejemplo, el alimento o la seguridad. En esta última puede estar relacionada a la seguridad del enjambre o la seguridad de la locación en la que se encuentra.
- **Principio de respuesta diversa:** los recursos encontrados por el enjambre no deben concentrarse en una región estrecha. La distribución debe diseñarse de modo que cada agente esté protegido al máximo frente a las fluctuaciones ambientales.
- **Principio de estabilidad:** se espera que el enjambre no cambie de manera rápida su comportamiento debido a las continuas fluctuaciones del entorno, ya que esto implica el uso de recursos y energía.
- **Principio de adaptabilidad:** es el lado opuesto del principio anterior, en el cual se espera que el enjambre pueda cambiar su comportamiento de acuerdo con las fluctuaciones del entorno, debido a la utilidad que esto le puede representar. La mejor respuesta es probablemente un equilibrio entre estos dos últimos principios, por lo que, la aleatoriedad juega un rol importante para estos algoritmos.

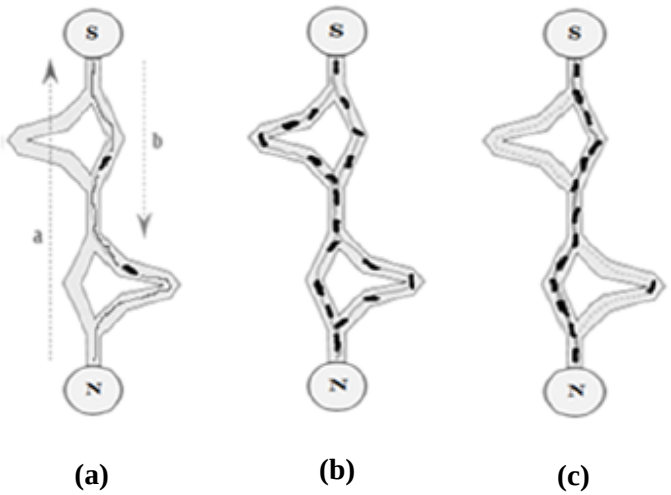
3.6.2 Algoritmos de inteligencia de enjambres

A continuación, se describirán los algoritmos de SI involucrados en el presente trabajo, los cuales han sido utilizados en tareas específicas dentro de la registración de imágenes, especialmente dentro del proceso de optimización. Además, han servido de inspiración para realizar algunos aportes.

3.6.2.1 Optimización por colonia de hormigas (ACO)

El primer modelo exitoso de inteligencia de enjambres es el algoritmo ACO, presentado por M. Dorigo [27], [34] en 1992, el cual ha sido utilizado en problemas de optimización discreta. ACO emula el comportamiento social de las colonias de hormigas. Estas constituyen uno de los ejemplos más representativos del comportamiento colectivo que se genera mediante la contribución individual. Las hormigas son insectos *casi ciegos* que buscan alimento para llevarlo a su nido.

El proceso empieza cuando las hormigas salen de su nido y se desplazan al azar en todas las direcciones. Una vez que una de ellas encuentra alimento regresa al nido dejando un rastro químico compuesto por *feromonas*. Otras hormigas pueden entonces detectar ese rastro y seguir el mismo camino. Lo interesante es que la frecuencia con que las hormigas visitan un camino está determinada por la concentración de feromonas. Dado que la feromona se evaporará naturalmente con el tiempo, la longitud del camino es un factor para considerar. Por lo tanto, como resultado surge *el camino más corto*, ya que las hormigas que siguen ese camino continúan agregando feromonas, lo que evita su evaporación [41]. Este proceso se resume en la **Fig. 3.5.**



(a) **(b)** **(c)**

Fig. 3.5: Optimización de la colonia de hormigas. **(a)** Marcar el rastro del camino descubierto entre el alimento y el nido. **(b)** Seguimiento del rastro. **(c)** Refuerzo del rastro con menor distancia. **Fuente:** https://es.wikipedia.org/wiki/Algoritmo_de_la_colonia_de_hormigas.

3.6.2.1.1 Metaheurística de ACO

ACO está definido con base en el rastro de feromonas durante su ubicación como su posterior seguimiento. Este algoritmo ha sido usado para la solución de varios problemas, entre ellos están el balanceo de la línea de ensamblaje [42], el problema del viajante (TSP[4]) [43] y la secuenciación del ADN [44].

La idea principal es modelar el problema para resolverlo como la búsqueda de una ruta óptima en un grafo ponderado, denominado grafo de construcción, y usar hormigas artificiales para buscar rutas de calidad [41]. En el grafo, las hormigas depositan de forma iterativa los rastros de feromonas para ayudar a elegir los nodos de los caminos de calidad que serán componentes de la solución. Las hormigas artificiales simulan el comportamiento de los reales en varios aspectos:

- Depositan rastros de feromonas en los nodos de caminos de calidad para reforzar aquellos más prometedores que posteriormente podrán ser parte de la solución.
- Construyen soluciones moviéndose a través del grafo y eligen sus caminos con respecto a las probabilidades que dependen de los rastros de feromonas previamente depositados.
- Los rastros de feromonas artificiales disminuyen lo suficientemente rápido en cada iteración, lo cual, simula el fenómeno real de evaporación observados en hormigas reales.

Un paso relevante para el algoritmo ACO es definir la función de aptitud o costo, con base en los componentes del grafo de construcción. Esta función determina la recompensa que obtendrá cada uno, basada en el rastro de feromonas, además de, establecer la forma en que las hormigas explotarán los componentes prometedores cuando construyan nuevas soluciones.

La función de costo será minimizada. Por tanto, las hormigas artificiales exploran el grafo y seleccionan los nodos que minimizan el costo general de la ruta de la solución. El **Algoritmo 3.1** resume la metaheurística de ACO [27].

[4] Por sus siglas en inglés de *Travelling Salesman Problem*.

Algoritmo 3.1: Metaheurística de ACO

---

1. Representar el espacio de soluciones mediante un grafo de construcción.
2. Establecer los parámetros de ACO y se inicializan los rastros de feromonas.
3. Generar las soluciones por hormigas a partir de cada exploración en el grafo guiado por el rastro de feromonas.
4. Actualizar la intensidad de las feromonas.
5. Ir al paso 3 y repetir hasta que se cumplan las condiciones de convergencia o terminación.

3.6.2.1.2 Generación de soluciones

El objetivo de ACO es construir soluciones de hormigas moviéndose estocásticamente a través de los nodos adyacentes en el grafo. Las hormigas son impulsadas por una regla de probabilidad (**Ec. 3.11**), denominada regla de elección de acción proporcional aleatoria o de transición de estado, para elegir secuencialmente los componentes de la solución basándose en las intensidades de las feromonas en la trayectoria y en la información heurística.

Una solución se construye cuando todos los componentes son seleccionados por una hormiga, es decir, cuando la hormiga ha completado un recorrido en el grafo. Una vez que se ha construido total o parcialmente una solución, la hormiga evalúa la solución que utilizará el siguiente paso para determinar la cantidad de feromona que se depositará [45], [46].

$$\mathrm{P}_{ij}^{k}(t) = \begin{cases} \dfrac{\left[\tau_{ij}(t)\right]^{\alpha} * [\eta_{ij}\,]^{\beta}}{\sum_{l \in \mathrm{N}_i^k} [\tau_{il}(t)]^{\alpha} * [\eta_{il}\,]^{\beta}}, & j \in \mathrm{N}_i^k \\ 0, & j \notin \mathrm{N}_i^k \end{cases} \tag{3.11}$$

donde $\mathrm{P}_{ij}^{k}(t)$ es la probabilidad de la hormiga k^{th}para moverse de un nodo i a un nodo j en la iteración o tiempo t, $\left[\tau_{ij}(t)\right]^{\alpha}$ es la cantidad de feromona que está conectando el nodo i con el nodo j, la cual está ponderada por la constante α y $\tau(t)$ es la información total de las feromonas contenida durante todo el proceso de búsqueda, $[\eta_{ij}\,]^{\beta}$ es el valor heurístico, o valor de visibilidad, que conecta los nodos (i, j), este valor es información conocida a priori sobre la definición del problema, N_i^k es el conjunto de nodos vecinos al nodo i^{th} de la hormiga k^{th}. $\mathrm{P}_{ij}^{k}(t) = 0$ si la hormiga no tiene permitido moverse a ninguno de los nodos vecinos. α y β son parámetros de ponderación que controlan la importancia relativa a la información de las feromonas vs. la heurística.

3.6.2.1.3 Actualización de las feromonas

Al inicio del proceso las feromonas se inicializan con un valor pequeño constante τ_0, pero una vez que las rutas de soluciones se establecen en el grafo de construcción, las feromonas se actualizan mediante la **Ec. 3.12.**

En el primer término se muestra, que todas las feromonas tienen un decremento en su valor de acuerdo con la *frecuencia de evaporación* ρ que permite que las hormigas no recorran rutas que no son óptimas. Además, la frecuencia de evaporación permite la *exploración* de nuevas regiones en el espacio de soluciones, evitando una convergencia prematura.

En el segundo término, las feromonas incrementan su valor en forma inversamente proporcional al costo de su ruta recorrida, es decir, proporcional a la calidad de la ruta. Esto permite que haya la *explotación*, de las rutas de calidad por el incremento de la probabilidad para ser elegidas otra vez por futuras hormigas [46].

$$\tau_{ij}(t+1) = (1-\rho) * \tau_{ij}(t) + \sum_{k=1}^{n} \Delta\tau_{ij}^{k}(t) \tag{3.12}$$

donde ij representan todos los nodos que conforman el grafo de construcción, $0 \leq \rho < 1$, n es el número total de hormigas y $\Delta\tau_{ij}^{k}(t)$ se calcula mediante la **Ec. 3.13.**

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} \dfrac{Q}{C^{k}(t)}, & si\ arc(i,j) \in \tau^{k}(t) \\ 0, & en\ cualquier\ otro\ caso \end{cases} \tag{3.13}$$

siendo Q es una constante específica de acuerdo con la aplicación, $C^{k}(t)$ son los valores totales de costos en el grafo de construcción, $\tau^{k}(t)$ es la ruta completada con la hormiga k^{th} en la iteración t y $arc(i,j)$ representa la conexión de los nodos (i,j) en la ruta establecida por la hormiga k^{th}.

3.6.2.2 Optimización por enjambre de partículas (PSO)

El segundo modelo exitoso de inteligencia de enjambres es el algoritmo PSO presentado por R. Eberhart y J. Kennedy [47] en 1995, el cual emula el comportamiento colectivo de las bandadas de aves en su búsqueda de alimento. PSO se basa en las variables físicas de velocidad y posición para simular el *"vuelo o desplazamiento"* de las partículas artificiales dentro del espacio de búsqueda del problema de optimización [48].

La observación de la naturaleza permite ver que las aves vuelan en grandes grupos, ya sea, en la búsqueda de alimento o en la migración de larga distancia. Por tanto, la visión es el sentido más importante para la organización de las bandadas [49]. Además, las aves tienen una interacción social eficiente que les permite: volar sin colisión incluso cuando pueden cambiar de dirección repentinamente, dispersarse y reagruparse rápidamente cuando reaccionan ante amenazas externas y evitar a los depredadores [46], [48].

PSO fue originalmente utilizado para resolver problemas de optimización continuos no lineales, pero posteriormente ha sido utilizado para resolver problemas con aplicación real, tales como: el seguimiento de sistemas dinámicos [50], la evolución de los pesos de una red neuronal artificial [51], la registración de imágenes médicas [52], [53], el aprendizaje de juegos [54] e incluso en la composición musical [55].

3.6.2.2.1 Metaheurística de PSO

El algoritmo PSO es una búsqueda basada en una población, donde cada miembro se denomina *partícula*, lo que permite encontrar soluciones óptimas usando dos variables: las posiciones en el espacio de soluciones y las velocidades que se ajustan dinámicamente de acuerdo con el historial de su rendimiento (**Fig. 3.6)**. PSO resuelve problemas donde las soluciones pueden ser representadas como un conjunto de puntos en el espacio de búsqueda $n - dimensional$. Desde esta perspectiva, cada partícula es una solución candidata al problema de optimización [56].

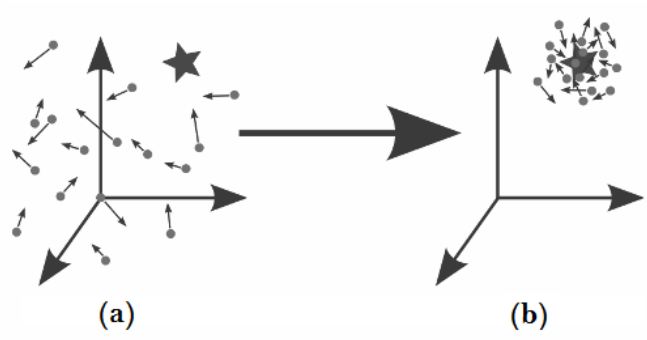


Fig. 3.6: Algoritmo PSO en un espacio de soluciones 3D. **(a)** Cada partícula del enjambre tiene asociada una posición y una velocidad para desplazarse en el espacio de búsqueda para llegar al óptimo global. **(b)** En el ultima iteración, el enjambre tenderá a converger en el óptimo global. **Fuente:** https://esa.github.io/pagmo2/docs/cpp/algorithms/pso.html.

El núcleo del algoritmo PSO clásico es el cálculo de la nueva posición para cada partícula de la población basada en la velocidad de desplazamiento. Cada partícula tiene una memoria para almacenar su mejor solución histórica, es decir, la mejor posición encontrada hasta un determinado instante. A esto se lo conoce como *experiencia.* Por tanto, el rendimiento exitoso radica en el intercambio continuo de experiencias de cada partícula con una parte o con todo el enjambre para conducirlo hacia áreas más prometedoras detectadas en el espacio de búsqueda.

En cada iteración, las partículas evalúan su posición actual con respecto a la función de aptitud del problema de optimización y comparan su valor actual con sus mejores posiciones históricas, pero también esta comparación se realiza de manera local o global *i.e.* dentro de su vecindario o con todo el enjambre.

Entonces, cada partícula actualiza su experiencia cuando la posición actual es mejor que la histórica y ajusta su velocidad para acercarse a la mejor partícula global del enjambre o a su vecino superior más cercano. Al final de cada iteración, la mejor partícula global del enjambre es actualizada si se ha encontrado una mejor posición en todo el enjambre [47].

Este proceso se resume en el **Algoritmo 3.2**, que muestra la secuencia en la ejecución del algoritmo PSO.

Algoritmo 3.2: Metaheurística de PSO

---

1. Inicializar el enjambre en el espacio de soluciones n-dimensional de manera aleatoria, con un valor de velocidad inicial.
2. Evaluar la función de aptitud deseada para actualizar las posiciones de las partículas.
3. Para cada partícula, actualizar su mejor posición local cuando la actual es mejor que la histórica.
4. Actualizar la mejor partícula global que tenga el mejor valor de aptitud del enjambre.
5. Actualizar las velocidades de todas las partículas usando la **Ec. 3.14.**
6. Actualizar las posiciones de cada partícula usando la **Ec. 3.15.**
7. Repetir los pasos 2 a 6 hasta que se cumpla el criterio de convergencia o de detención.

3.6.2.2.2 Actualización de velocidad y posición

PSO se concibió originalmente como un algoritmo de optimización global, por lo tanto, cada partícula compara su valor de aptitud con todo el enjambre y ajusta su velocidad en dirección a la

mejor partícula global encontrada. La actualización de la velocidad en el algoritmo PSO original se realiza mediante la **Ec. 3.14.**

$$v_i^d(t+1) = v_i^d(t) + c_1 r_1 \left(P_i^d(t) - x_i^d(t)\right) + c_2 r_2 \left(p_g^d(t) - x_i^d(t)\right) \quad (3.14)$$

donde v_i^d representa la velocidad de la partícula i^{th} en la dimensión d^{th}, t es el tiempo o el número de iteración, P_i^d representa la mejor posición local encontrada por la partícula i^{th}, x_i^d representa la posición actual de la partícula i^{th}, c_1 y c_2 son valores constantes positivos denominados parámetros *cognitivos o sociales*, r_1 y r_2 son números aleatorios con distribución uniforme en el rango $[0.0, 1.0]$, los cuales introducen la aleatoriedad en la estrategia de búsqueda y p_g^d es la posición de la mejor partícula global encontrada en el instante t en la dimensión d^{th}.

Una vez que se ha calculado la velocidad, es decir, el valor que se desplazará cada partícula en cada una de las dimensiones del espacio de soluciones de la función se procede a actualizar las nuevas posiciones siguiendo la **Ec. 3.15.**

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1) \quad (3.15)$$

A lo largo de los años se han presentado modificaciones al algoritmo original PSO, lo cual, lo ha hecho mucho más sofisticado y con mejor rendimiento para la optimización estocástica en problemas con un único y múltiples objetivos [57], [58].

3.6.2.3 Optimización por colonia de abejas artificiales (ABC)

Otro de los modelos exitosos, inspirado en colmenas de abejas, fue presentado por D. Karaboga y B. Basturk [59], [60] en 2007, el cual está basado en el comportamiento de las abejas durante la búsqueda de alimento.

Este comportamiento se puede resumir de la siguiente manera: al inicio, se envían algunas abejas para buscar fuentes de alimentos prometedoras. Una vez que se ha ubicado una buena fuente de alimento, las abejas regresan a la colmena y realizan una danza de movimiento para difundir la información encontrada. Esta información incluye: la distancia, la dirección y la calidad de dicha fuente. Cuanto mejor sea su calidad más abejas serán atraídas y entonces, la mejor fuente de alimento es descubierta [61].

Los algoritmos de optimización ABC han sido aplicados en optimización matemática, donde se pretende encontrar soluciones globales óptimas a determinadas funciones. Además, se pueden aplicar en problemas de entrenamiento de redes neuronales artificiales [62], [63] y en el procesamiento de imágenes [64], [65].

3.6.2.3.1 Metaheurística de ABC

El algoritmo ABC está definido a partir de 3 grupos de abejas: las exploradoras, las empleadas y las espectadoras. Se puede decir que una colmena de abejas está dividida en mitad abejas empleadas y mitad abejas espectadoras.

Las abejas empleadas descubren nuevas fuentes de alimento y llevan la información para compartirla con las abejas espectadoras que están en expectativa de dicha información. Esta se comparte bailando en un área designada dentro de la colmena. La naturaleza del baile es proporcional al contenido de néctar de la fuente encontrada. Las abejas miran el baile y eligen

una determinada fuente acorde con la probabilidad proporcional a la calidad de dicha fuente, es decir, las buenas fuentes atraen a más abejas (**Fig. 3.7**).

Cada vez que se explota completamente una fuente, todas las abejas empleadas asociadas con ella, la abandonan y se convierten en exploradoras. La exploración se realiza de manera aleatoria. Por tanto, las abejas exploradoras realizan el trabajo de *exploración*, mientras que, las abejas empleadas y espectadoras el trabajo de *explotación* [60].

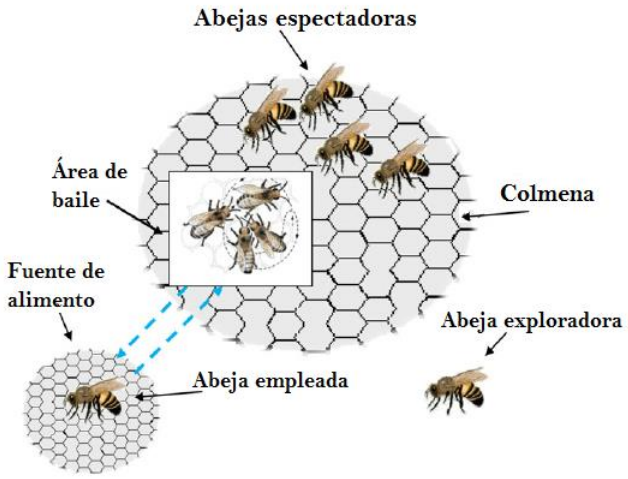


Fig. 3.7: Representación del algoritmo ABC. Descripción de los grupos de abejas que forman parte del procedimiento de optimización.

El proceso de ABC se resume en el **Algoritmo 3.3**, que muestra la metaheurística asociada a él.

Algoritmo 3.3: Metaheurística de ABC

---

1. Inicializar las posiciones de las abejas empleadas en el espacio de soluciones n-dimensional de manera aleatoria. Una fuente de alimento está asociada a una abeja empleada.
2. Determinar la cantidad de "néctar" en la fuente de alimento.
3. Mover las abejas espectadoras de acuerdo con la probabilidad de selección de la fuente de alimento. (**Ec. 3.16)**
4. Explotar la fuente de alimento. (**Ec. 3.17)**
5. Almacenar la información de la mejor fuente de alimento encontrada.
6. Buscar nuevas fuentes de alimento, convirtiendo a las abejas empleadas en exploradoras. (**Ec. 3.18)**
7. Repetir los pasos 2 a 6 hasta que se cumpla el criterio de convergencia o de detención.

3.6.2.3.2 Selección de las fuentes de alimentos

Para el algoritmo ABC, la posición de una fuente de alimento representa una solución candidata al problema de optimización y su cantidad de "néctar" corresponde a la calidad de la función de aptitud asociada a dicha solución. Una abeja espectadora elige una fuente de alimento mediante el valor de probabilidad p_i asociado. Esta probabilidad se calcula mediante la **Ec. 3.16.**

$$p_i = \frac{fit_i}{\sum_{i=1}^{n} fit_n} \tag{3.16}$$

donde fit es el valor de la función de aptitud de la i-ésima abeja empleada, la cual representa una solución i y n es el número total de abejas que conforman la colmena.

Para explotar la fuente de alimento, las abejas espectadoras analizan el vecindario de la fuente de alimento encontrada por la abeja empleada. Esta explotación está basada en la **Ec. 3.17.**

$$v_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{kj}) \tag{3.17}$$

siendo j un valor elegido aleatoriamente del conjunto de dimensiones de la función $j \in \{1,2,..,D\}$, donde D es la dimensión del problema a optimizar, x_{ij} es la posición de la i-ésima abeja en la dimensión j, k es un valor elegido aleatoriamente del conjunto total de abejas de la colmena, por tanto, $k \in \{1,2,..,n\}$ y ϕ_{ij} es un número aleatorio uniforme en el rango $[-1,1]$.

3.6.2.3.3 Actualización de posiciones

Si las soluciones encontradas por las abejas espectadoras mejoran la función de aptitud de la solución encontrada por la abeja empleadas, entonces esta actualiza su memoria con el nuevo valor encontrado.

Cuando una fuente de alimento es abandonada por las abejas, esta se reemplaza por una nueva fuente descubierta por las exploradoras. En ABC, esto se emula produciendo una posición al azar y reemplazándola por la posición abandonada. Esto ocurre cuando no se puede mejorar una posición a través de un número determinado de ciclos [59]. Las nuevas posiciones se aleatorizan en el espacio de soluciones, mediante la **Ec. 3.18.**

$$x_i^j = x_{min}^j + rand(0,1)(x_{max}^j - x_{min}^j) \tag{3.18}$$

De igual manera, para este algoritmo hay varias versiones que lo hacen más sofisticado y adaptable a diferentes problemas de optimización [66], [67].

3.6.2.4 Algoritmo de fuegos artificiales para optimización (FWA)

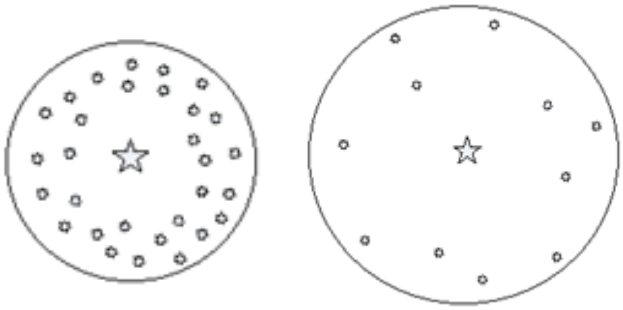
Este modelo está inspirado en la observación de las explosiones de fuegos artificiales en el cielo, propuesto por Y. Tan et al. [68] y es aplicado en la optimización global de funciones complejas. Cuando un fuego artificial explota, una lluvia de chispas se dispersa en el espacio local alrededor del él. Este proceso se puede ver como una búsqueda, mediante las chispas generadas, en el espacio local alrededor del punto específico donde se dispara el fuego artificial durante la explosión (**Fig. 3.8**).

(a) **(b)**

Fig. 3.8: Modelado del algoritmo FWA. **(a)** Explosión real de un fuego artificial en el cielo. **(b)** Representación de un fuego artificial en un espacio de búsqueda al generar una determinada cantidad de chispas en una región local a su alrededor.

3.6.2.4.1 Metaheurística de FWA

Se puede considerar una buena explosión en un fuego artificial real cuando la cantidad de chispas es alta y estas se dispersan a una distancia adecuada. En el algoritmo FWA, se considera un buen fuego artificial cuando está ubicado en un área prometedora, es decir, cerca de la ubicación óptima. Por lo tanto, se utilizan más chispas en un área menor para buscar en el área alrededor del él. En contraste, un fuego artificial malo puede estar lejos de la ubicación óptima, por tanto, el radio de búsqueda debe ser mayor. La cantidad de chispas se generan de acuerdo con el valor de su función de aptitud del problema de optimización [70] (**Fig. 3.9**).



(a) **(b)**

Fig. 3.9: Tipo de explosiones en FWA. **(a)** Fuego artificial en un área prometedora genera buena explosión. **(b)** Cuando la región no está cerca del óptimo se genera una mala explosión. **Fuente:** Y. Tan et al., 2010.

El proceso de FWA se resume en el **Algoritmo 3.4**, que muestra la metaheurística asociada a él.

Algoritmo 3.4: Metaheurística de FWA

---

1. Inicializar aleatoriamente n posiciones para los fuegos artificiales.
2. Evaluar las posiciones obtenidas y hacerlas explotar obteniendo el número de chispas y las distancias de explosión para cada uno de ellos. (**Ec. 3.19 y 3.20)**
3. Seleccionar aleatoriamente un número de fuegos artificiales para generar chispas uniformes y gaussianas. (**Ec. 3.21 y 3.22)**
4. Seleccionar la mejor localización encontrada y mantenerla en la siguiente generación.
5. Seleccionar aleatoriamente $n-1$ localizaciones, tanto del grupo de los fuegos artificiales como de las chispas, de acuerdo con su función de probabilidad. (**Ec. 3.23)**
6. Repetir los pasos 2 a 5 hasta que se cumpla el criterio de convergencia o de detención.

3.6.2.4.2 Número de chispas y distancia de explosión

El número de chispas que genera cada fuego artificial depende del valor de su función de aptitud y está definido como lo muestra la **Ec. 3.19** [68], [69]:

$$s_i = m \cdot \frac{y_{max} - f(x_i) + \varepsilon}{\sum_{i=1}^{n}\left(y_{max} - f(x_i)\right) + \varepsilon} \tag{3.19}$$

donde s es el número de chispas para el fuego artificial i^{th} para $i = (1, .., n)$, m es un parámetro de control, y_{max} es el peor valor de la función objetivo encontrado, es decir, si se considera un problema de minimización, $y_{max} = \max(f(x_i))$ y ε es la constante épsilon para evitar una división por cero.

La distancia de la explosión está definida mediante la **Ec. 3.20**:

$$A_i = \hat{A} \cdot \frac{f(x_i) - y_{min} + \varepsilon}{\sum_{i=1}^{n}(f(x_i) - y_{min}) + \varepsilon} \tag{3.20}$$

donde $\hat{A}$ es una constante de control que determina la distancia máxima aceptada, y_{min} es el mejor valor de la función objetivo encontrado, si es un problema de minimización $y_{min} = \min(f(x_i))$.

3.6.2.4.3 Chispas uniformes y gaussianas

Cada fuego artificial i genera $\widehat{s_i}$ chipas. Para esto, se genera primero un valor aleatorio de acuerdo con la dimensión d de la función objetivo, definido por $z = d * rand(0,1)$, que se utiliza para elegir z componentes de la dimensión total. La ubicación de la chispa se calcula usando la **Ec. 3.21**:

$$x_k^i = x_k^i + h \tag{3.21}$$

donde $h = A_i * rand(-1,1)$ y es el valor de desplazamiento, $rand$ es un número aleatorio de distribución uniforme, k es la dimensión seleccionada a partir de z y i corresponde al i^{th} fuego artificial.

Una vez que todos los fuegos artificiales han generado el número total de chispas correspondientes, se eligen de manera aleatoria un número de ese conjunto total para generar chispas gaussianas. Este paso tiene la finalidad de dar mayor robustez al algoritmo. De manera similar a lo anterior, se calcula el valor z y la nueva ubicación está definida mediante la **Ec. 3.22**:

$$x_k^j = x_k^j + g \tag{3.22}$$

donde $g = gauss(1,1)$ y es el valor de desplazamiento, $gauss$ genera un número aleatorio con distribución gaussiana de $\mu = 1$ y $\sigma = 1$, k es la dimensión seleccionada a partir de z y j corresponde al j^{th} fuego artificial seleccionado del conjunto total obtenido hasta el momento.

3.6.2.4.4 Selección por función de probabilidad

Del conjunto total de fuegos artificiales y chispas tanto uniformes como gaussianas, se eligen las $n - 1$ localizaciones para la siguiente generación, la cual depende de la función de probabilidad presentada en la **Ec. 3.23**:

$$p(x_i) = \frac{R(x_i)}{\sum_{j \epsilon K} R(x_j)} \tag{3.23}$$

donde $R(x_i) = \sum_{j \epsilon K} \|x_i - x_j\|$ es una métrica de distancia (Euclídea, Manhattan, etc.), y K es el conjunto de fuegos artificiales y chispas.

De igual manera se han presentado algunas modificaciones a este algoritmo, lo que ha permitido un mejor rendimiento en distintos problemas de optimización [70], [71].

3.7 Contribuciones originales

Todas las contribuciones que se muestran a continuación se han realizado utilizando imágenes médicas. La registración se ha realizado en monomodo y multimodo utilizando imágenes MRI de cerebro para las deformaciones rígidas.

3.7.1 Definición del problema de registración rígida

Como se ha mencionado, la registración de imágenes es un problema de optimización, donde se busca maximizar una medida de similaridad o en su defecto minimizar el error entre la imagen de referencia y la imagen móvil. Para esto es necesario calcular una transformación geométrica que se aplicará a la imagen móvil para llevar al espacio dimensional de la imagen de referencia.

Formalmente, se puede definir de la siguiente manera: Sean I, J las imágenes de referencia y móvil respectivamente, $I(x, y)$ y $J(u, v)$ representan sus valores de intensidad donde $\{x, y\} \in \Omega_I$ y $\{u, v\} \in \Omega_J$.

Por lo tanto, las imágenes son funciones definidas: $I(x, y)$: $\Omega_I \subseteq \mathbb{R}^2 \rightarrow \mathbb{R}$, $J(u, v)$: $\Omega_J \subseteq \mathbb{R}^2 \rightarrow \mathbb{R}$, donde se pretende calcular una transformación geométrica $\mathcal{T}_\phi$: $\mathbb{R}^2 \rightarrow \mathbb{R}^2$ donde ϕ es el vector de parámetros de la transformación, en registración rígida son: escala, rotación y traslación; entonces:

$$\mathcal{T}, \phi, J(u, v)\colon\ (u, v;\ \phi) \rightarrow \mathcal{T}_\phi(u, v)$$

Esto permite obtener una nueva imagen $R = \mathcal{T}_\phi(J)$ y mediante el proceso de registración se tiene como función objetivo $min\ (\|I(x, y) - R(u', v')\|)$.

3.7.2 Comparación de algoritmos aplicados en registración rígida

Inicialmente se realizó una comparación del rendimiento entre los algoritmos de inteligencia de enjambres mencionados en la sección anterior, al aplicarse en la registración rígida de imágenes médicas. El tipo de registración utilizada está basada en la *intensidad de los píxeles* [4], [52].

El aporte de esta sección es la evaluación del rendimiento de los algoritmos de optimización aplicados en esta problemática.

Datos de las pruebas

Para las pruebas se realizaron 15 ejecuciones independientes de cada algoritmo, el número de iteraciones es 50 y el número de partículas es 20.

Imágenes MRI para registración

Las imágenes que se usaron para evaluar el rendimiento del algoritmo propuesto se muestran en la **Fig. 3.10.** Su tamaño es de 256×256 con 8 bits por cada canal, tanto en las imágenes en escala de grises (secuencia T1 y T2), como para la imagen de color (SPECT).

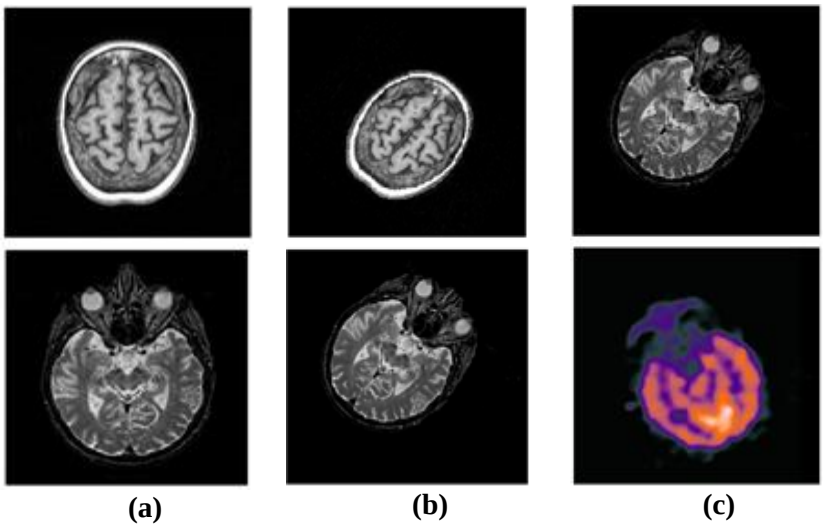


Fig. 3.10: Imágenes MRI cerebrales en secuencias T1, T2 y modalidad SPECT. **(a)** Imágenes de referencia. **(b)** Imágenes móviles utilizadas para registración monomodal. **(c)** Imágenes móviles utilizadas para registración multimodal.

El análisis del rendimiento está basado en el promedio de la medida de similaridad alcanzada durante las ejecuciones. Además, se muestra la distribución estadística de los datos resultantes, lo cual se utiliza para analizar la estabilidad del algoritmo en términos de valor medio y desviación estándar.

Registración monomodal

La registración monomodal se realiza entre imágenes de la misma modalidad T1-T1 y T2-T2. La **Fig. 3.11** muestra el promedio de la *correlación de Pearson* alcanzado por cada algoritmo durante las iteraciones por cada prueba. Además, se muestra la gráfica de la distribución de los datos de los resultados obtenidos, los cuales indican la simetría en la distribución resultante y la dispersión de los puntos respecto de la mediana.

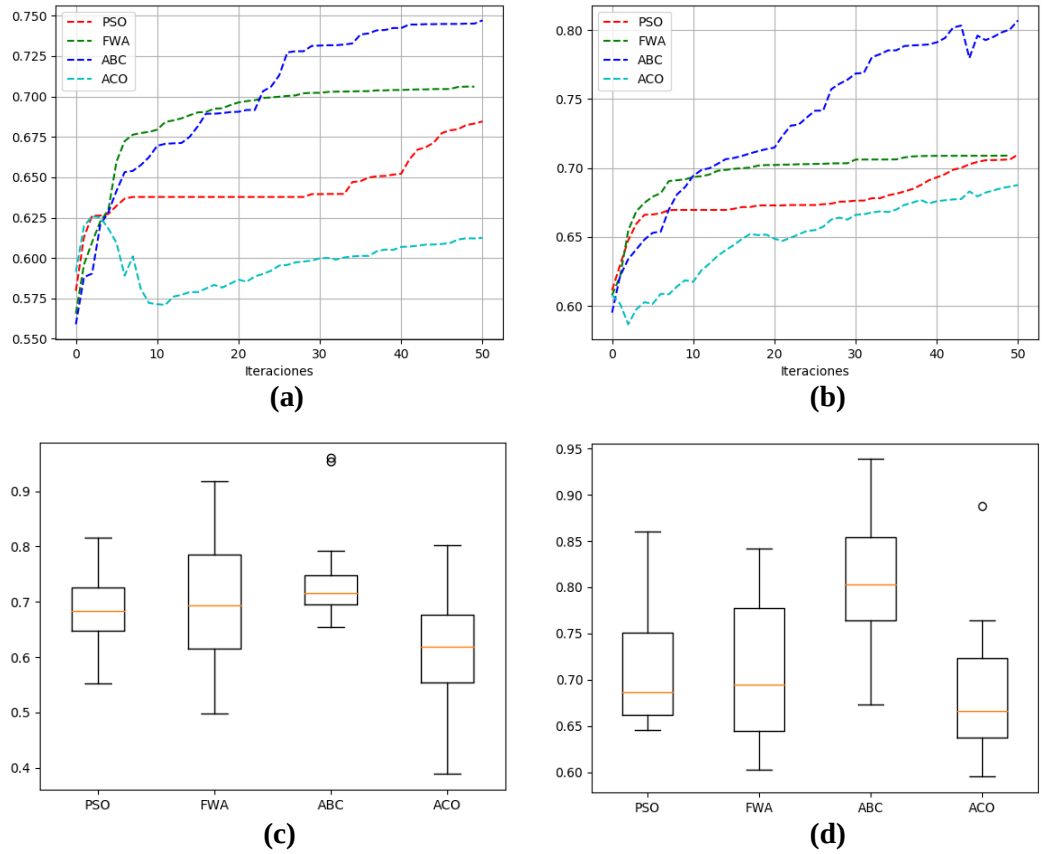


Fig. 3.11: Gráficas del promedio de la correlación de Pearson por cada algoritmo. **(a) Prueba 1** entre las imágenes 1 y 2 superiores de Fig. 3.10 **(b) Prueba 2** entre las imágenes inferiores 1 y 2 de Fig. 3.10. **(c)** Gráfica de la distribución estadística de los resultados de la prueba 1. **(d)** Gráfica de la distribución estadística de los resultados de la prueba 2.

En las **Tabla 3.1 y 3.2**, se muestran los valores más representativos obtenidos por cada algoritmo en las pruebas monomodales 1 y 2.

Tabla 3.1: Valores de correlación de Pearson obtenidos en la prueba 1.

Algoritmo	**Promedio *valor de correlación***	**Desviación estándar**	**Max. *valor de correlación***
PSO	0.69	0.067	0.82
FWA	0.71	0.113	0.92
ABC	0.75	0.089	0.96
ACO	0.61	0.112	0.80

Tabla 3.2: Valores de correlación de Pearson obtenidos en la prueba 2.

Algoritmo	**Promedio *valor de correlación***	**Desviación estándar**	**Max. *valor de correlación***
PSO	0.71	0.057	0.86
FWA	0.71	0.074	0.84
ABC	0.81	0.067	0.94
ACO	0.69	0.071	0.89

Registración multimodal

La registración multimodal se realiza entre imágenes de distinta modalidad T1-T2 y T2-SPECT. La **Fig. 3.12** muestra el promedio de la *información mutua* alcanzado por cada algoritmo durante las iteraciones por cada prueba, además, se muestra la gráfica de la distribución de los datos de los resultados obtenidos.

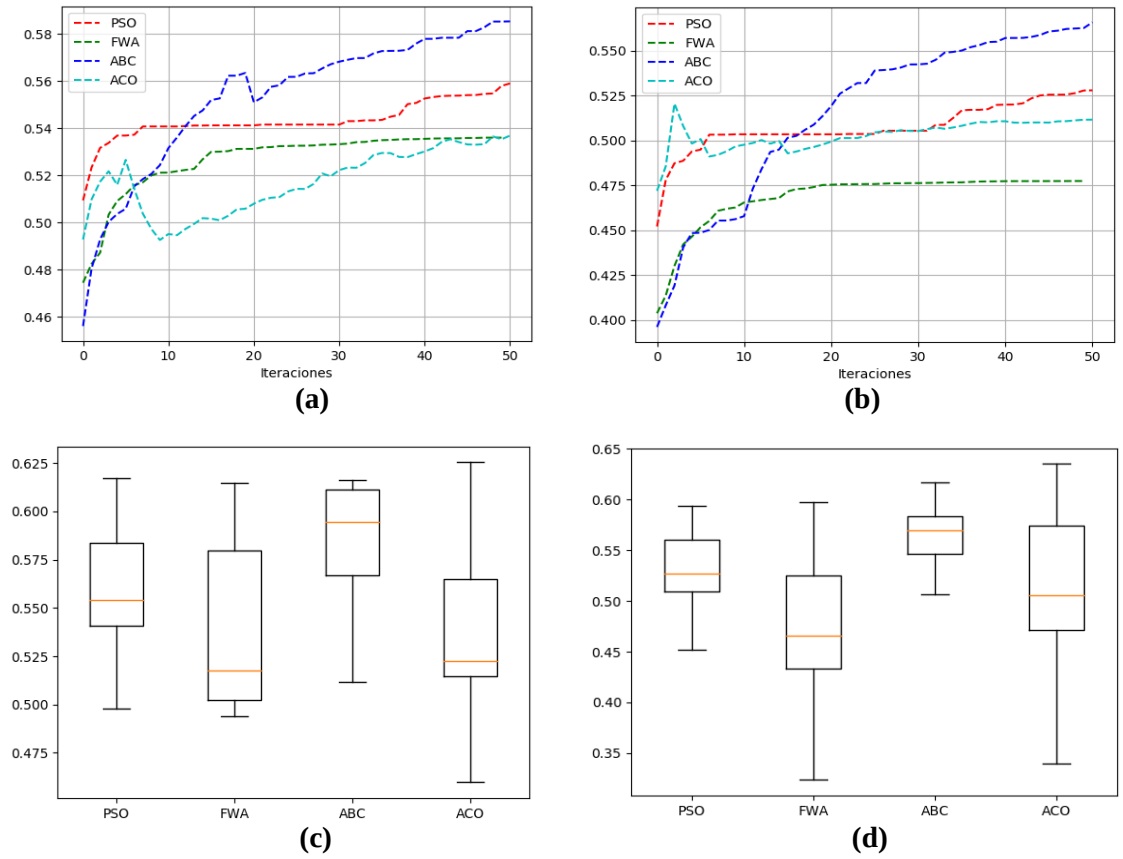


Fig. 3.12: Gráficas del promedio de la información mutua por cada algoritmo. **(a) Prueba 3** entre las imágenes 1 y 3 superiores de Fig. 3.10 **(b) Prueba 4** entre las imágenes inferiores 1 y 3 de Fig. 3.10. **(c)** Gráfica de la distribución estadística de los resultados de la prueba 3. **(d)** Gráfica de la distribución estadística de los resultados de la prueba 4.

En las **Tabla 3.3 y 3.4** se muestran los valores más representativos obtenidos por cada algoritmo en las pruebas multimodales 3 y 4.

Tabla 3.3: Valores de información mutua obtenidos en la prueba 3.

Algoritmo	**Promedio *valor de MI***	**Desviación estándar**	**Max. *valor de MI***
PSO	0.56	0.034	0.62
FWA	0.54	0.042	0.62
ABC	0.59	0.031	0.62
ACO	0.54	0.040	0.63

Tabla 3.4: Valores de información mutua obtenidos en la prueba 4.

Algoritmo	**Promedio *valor de MI***	**Desviación estándar**	**Max. *valor de MI***
PSO	0.53	0.040	0.59
FWA	0.48	0.070	0.60
ABC	0.57	0.030	0.62
ACO	0.51	0.084	0.64

Conclusiones a partir del análisis de los resultados

De acuerdo con los resultados obtenidos, todos los algoritmos de inteligencia de enjambres utilizados en esta comparativa aportan un resultado adecuado a la registración. Esto debido a que, en todos los casos, los algoritmos maximizan la medida de similaridad durante sus procesos.

Sin embargo, el algoritmo ABC (Optimización por colonia de abejas) basado en [66], es el mejor rendimiento presenta frente al problema de registración rígida, por su rendimiento al maximizar la función objetivo, ya que su valor promedio supera en 0.60 de correlación. Además, tiene mayor estabilidad en la distribución estadística de sus resultados.

3.7.3 Propuesta para mejorar el rendimiento de PSO en registración

Este aporte se presenta en el artículo titulado “Improved Particle Swarm Optimization algorithm applied to rigid registration in medical images” [72], el cual tiene como base una variante del algoritmo PSO denominada *de peso decreciente* [57], [58].

En esta variante de PSO, el coeficiente de inercia va decreciendo de manera proporcional al incremento de iteraciones, lo cual, permite estabilizar la solución óptima encontrada hacia el final del proceso.

Metaheurística del algoritmo propuesto

El aporte principal de esta sección es que las partículas no se inicializan aleatoriamente en el espacio de soluciones como lo hace normalmente PSO, sino que, se realiza un procedimiento inicial que sirve de guía para la búsqueda, además el espacio de soluciones se divide en subregiones, lo que permite mejorar el tiempo y el rendimiento del algoritmo.

Esta propuesta se resume en el **Algoritmo 3.5**.

Algoritmo 3.5: Metaheurística de PSO mejorado para registración

---

1. Calcular el valor inicial de similaridad entre la imagen de referencia y la móvil. Si es registración monomodal se utiliza la *correlación de Pearson* y en multimodal la *información mutua.*
2. Aleatorizar partículas en el espacio de soluciones hasta que se encuentre una que mejore el valor inicial de similaridad.
3. Con la partícula obtenida, dividir el espacio de soluciones en subregiones, donde se realizará una búsqueda localizada.
4. Ejecutar PSO en las subregiones del espacio de soluciones.
5. Aleatorizar las partículas en el espacio de soluciones global, cada cierto número de iteraciones, para evitar caer en mínimos locales
6. Finalizar el algoritmo cuando se ha alcanzado el número de iteraciones definido.

Datos de las pruebas

Para las pruebas se utilizó un arreglo que contenía diferentes valores para el número de partículas utilizadas por PSO. Esto con el fin de determinar el número óptimo con el cual el algoritmo pueda obtener el mejor rendimiento. Los valores están en el rango $[20, 60]$. El número de iteraciones es 60 y se realizaron 4 pruebas (dos por cada modalidad de registración) donde los resultados se comparan entre la variante de PSO y el algoritmo PSO propuesto.

Resultados obtenidos

Los resultados se evalúan en referencia al valor de la función de aptitud obtenida: correlación de Pearson e Información Mutua entre las imágenes de referencia y las registradas, el tiempo de procesamiento obtenido y el error cuadrático medio entre las imágenes de salida. A continuación, se muestran los resultados obtenidos respecto de la medida de similaridad, pues son los que aportan la información específica del rendimiento de cada algoritmo[5].

En la **Fig. 3.13,** se muestran los valores obtenidos en las dos pruebas de registración monomodal T1-T1 y T2-T2, tanto para el algoritmo PSO original como para el propuesto.

[5] Ver referencia [72] para verificar los resultados completos.

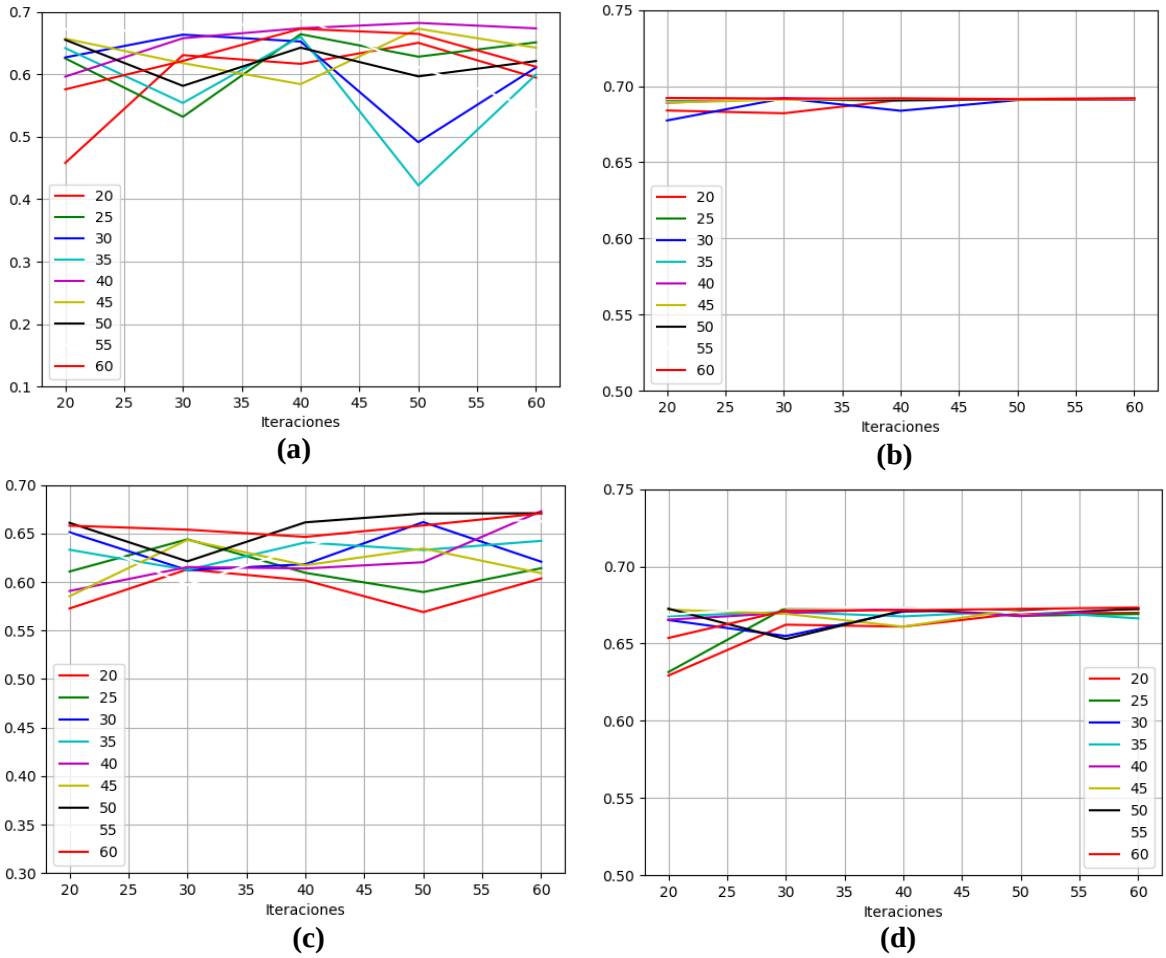


Fig. 3.13: Valor de correlación de Pearson obtenida con cada número de partículas. **(a)** PSO original con imagen de referencia y móvil T1. **(b)** PSO propuesto con imágenes T1. **(c)** PSO original con imagen de referencia y móvil T2. **(d)** PSO propuesto con imágenes T2.

En la **Fig. 3.14,** se muestran los valores obtenidos en las dos pruebas de registración multimodales T1-SPECT y T2-SPECT, tanto para el algoritmo PSO original como para el propuesto.

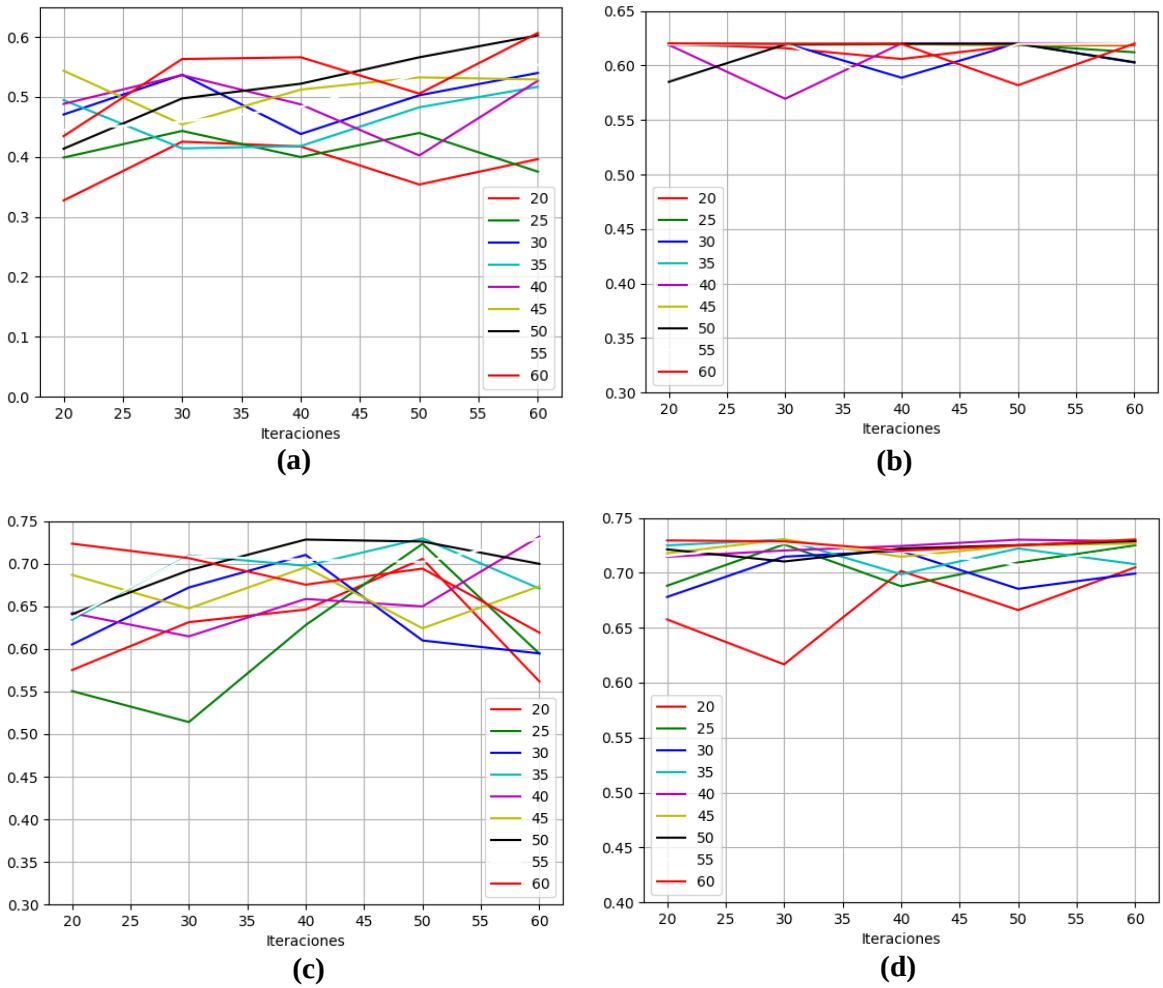


Fig. 3.14: Valor de información mutua obtenida con cada número de partículas. **(a)** PSO original con imágenes T1 - SPECT **(b)** PSO propuesto con imágenes indicadas **(c)** PSO original con imágenes T2 - SPECT **(d)** PSO propuesto con imágenes indicadas.

Conclusiones a partir del análisis de los resultados

De acuerdo con los resultados obtenidos el algoritmo PSO propuesto mejora notablemente el rendimiento respecto del algoritmo original, al aplicarse en registración monomodo y multimodo. Los resultados del error son menores al 0.1% con el algoritmo propuesto.

Además, por el valor de la medida de similaridad obtenida se concluye que el número de partículas a partir de 30 no tiene gran relevancia para el rendimiento del algoritmo. Esto no sucede en el caso de PSO original. Los valores de similaridad en promedio se encuentran en el rango de $[0.65 - 0.70]$.

El algoritmo propuesto no requiere un gran número de partículas para obtener buenos resultados, por lo tanto, permite optimizar el tiempo y el uso de los recursos computacionales disponibles.

3.7.4 Propuesta del algoritmo LiA para optimización global

El aporte de esta sección es la propuesta de un nuevo algoritmo de inteligencia colectiva inspirado en la naturaleza, llamado algoritmo de rayos (LiA) para optimización global unimodal y multimodal[6], inspirado en la forma y el fenómeno físico de los rayos. El conjunto de soluciones en el espacio de búsqueda se genera siguiendo la forma de una descarga eléctrica atmosférica.

Presentación del algoritmo

Se inicializan puntos (soluciones candidatas) en el espacio de búsqueda de la función objetivo. Para esto se utiliza una inicialización aleatoria llamada *destellos*, la cual está inspirada en los destellos de luz previos a una tormenta. La inicialización aleatoria es utilizada por la mayoría de los algoritmos de inteligencia colectiva por su rapidez especialmente en funciones con grandes dimensiones. Sin embargo, la inicialización es un paso de gran importancia para los algoritmos estocásticos, ya que de ésta depende obtener una buena convergencia.

A la población inicial se la denomina *Puntas líder artificiales*, las cuales, están asociadas a una carga negativa según el modelo real. A partir de cada una de ellas, se generan dos tipos de ramificaciones: la rama principal, *para exploración*, que es una ruta generada por una secuencia de soluciones candidatas separadas una distancia basado en un valor de paso orientadas hacia el punto mínimo global encontrado en la iteración actual; y, las ramas secundarias, *para explotación*, que genera una secuencia de soluciones a partir de los puntos de la rama principal, a través de una selección probabilística, de acuerdo con el valor de la función de aptitud obtenida con dirección aleatoria.

Para generar la forma de zigzag típica de un rayo, se establece un proceso denominado *mutación*, donde mediante una probabilidad de selección por cada punto en las dos ramas. Si la probabilidad es mayor a 0.5, la posición del punto será modificada mediante la suma con un número aleatorio normal con (μ, σ). Esto permite que el algoritmo tenga mayor robustez para evitar mínimos locales y mejorar los resultados en la optimización de funciones.

Definiciones de LiA

En el contexto de este trabajo, el espacio de búsqueda de una función (el dominio de parámetros) es un conjunto de regiones "cargadas eléctricamente" en las que se puede generar un rayo. Físicamente, un rayo se desplaza desde una carga negativa hasta encontrar una carga positiva con la cual cierra el circuito eléctrico [73], [74].

LiA empieza con *n* ubicaciones inicializadas de una manera novedosa en el espacio de búsqueda. Estas ubicaciones iniciales se consideran "cargas negativas o leader tip". Las mejores y peores ubicaciones se obtienen de este conjunto. Estos valores se utilizan para calcular el número de puntos o líderes artificiales que formarán parte de cada rayo. Además, las mejores ubicaciones se utilizan para guiar en la búsqueda del óptimo global. El conjunto de puntos de un rayo se divide en dos categorías: *rama principal y ramas secundarias.*

Cuando un rayo completa su trayectoria, se obtiene la ubicación del punto que mejora el valor de la función de aptitud del punto inicial, a la cual se denomina una "carga positiva" de acuerdo con el modelo de un rayo real. Esto se realiza en los *n* relámpagos donde las “cargas positivas” son

[6] Se conoce como unimodal a las funciones objetivo que tienen un solo máximo o mínimo global, es decir, solo hay un objetivo como resultado. Las funciones multimodales, por el contrario, tienen dos o más objetivos como resultado para el problema de optimización.

llamadas *las mejores posiciones locales*. Finalmente, mientras el número de iteraciones está disponible, las nuevas n posiciones para las "cargas negativas" se actualizan a través de las mejores posiciones locales y la mejor posición global. El modelo generado de rayo generado por LiA se presenta en la **Fig. 3.15.**

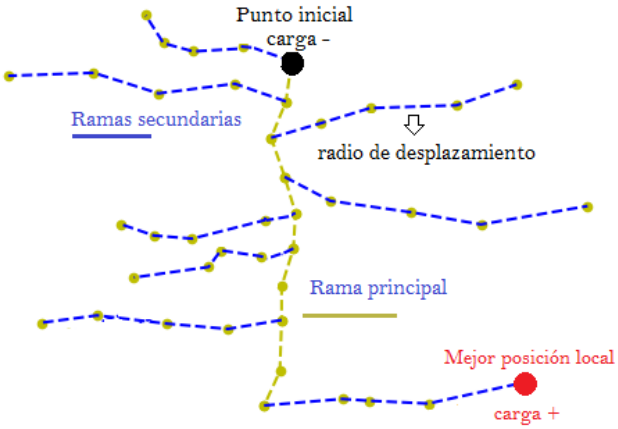


Fig. 3.15: Forma del "rayo" generado por LiA para optimización junto con sus componentes.

Inicialización de destellos

En algoritmos de inteligencia de enjambre o inteligencia colectiva, la población inicial es una tarea crucial para lograr una óptima convergencia en términos de tiempo de procesamiento y calidad de la solución final. La inicialización aleatoria es la más usada en esta clase de algoritmos.

En [66] se propone una novedosa inicialización de tipo caótico, la cual mejora la diversidad de la población inicial para el algoritmo ABC; sin embargo, cuando se trata de funciones objetivo de alta dimensionalidad, esta inicialización tiende a ralentizar el proceso de optimización. Por tanto, LiA utiliza una inicialización novedosa basada en la inicialización aleatoria, la cual se muestra en el **Algoritmo 3.6.**

Algoritmo 3.6: Inicialización de destellos.

Con el valor n para el conjunto de cargas iniciales, $lower\_bound$ (límite inferior) y $upper\_bound$ (límite superior) y D la dimensión de la función objetivo.

- $gr = []$
- **for** i **in range** (m)**:**
- $\quad X = lower_{bound} + (upper_{bound} - lower_{bound}) * rand(n, D)$
- $\quad gr.append(X)$
- $x_{fit} = fitness(gr)$
- **for** i **in range** (rep)**:**
- $\quad X\_ = permutación(gr)$
- Calcular el valor de diversidad de X_ (**Ec. 3.24)**
- Calcular el promedio del valor de aptitud de X_
- Seleccionar la mejor combinación con el valor más alto de diversidad y menor promedio de la función de aptitud.

La idea consiste en inicializar m poblaciones aleatorias y agrupar cada una de las partículas en grupos distintos para finalmente seleccionar aquella población que tenga mayor diversidad y el

menor promedio de la función de aptitud, lo cual permite que LiA empiece con una población adecuada para el proceso. La diversidad de la población se calcula mediante la **Ec. 3.24:**

$$Diversidad = \frac{1}{n}\sum_{i=1}^{n}\sqrt{\frac{1}{D}\sum_{j=1}^{D}(x_{i,j} - \overline{x_j})^2} \tag{3.24}$$

Cálculo del número de puntos por cada rayo

El total de puntos que formarán la rama principal de cada rayo se calcula mediante la **Ec. 3.25.**

$$PB_i = p_{min} + \left(\frac{f(x_i) - f_{min}}{f_{max} + f_{min} + \varepsilon}\right) * (p_{max} - p_{min}) \tag{3.25}$$

donde $p_{max}\ y\ p_{min}$ son hiperparámetros constantes que limitan la cantidad de puntos que puede tener un rayo. f_{min} es el peor valor y f_{max} el mejor valor de la función de aptitud encontrada en el conjunto de puntos iniciales. Si se está minimizando una función, f_{min} será el punto con el valor de la función de aptitud más grande y f_{max} el punto con el valor más pequeño encontrado en todo el conjunto inicial. $f(x_i)$ es el valor de la función de aptitud para el punto $i = (1,2,..,n)$ y ε es el valor épsilon para evitar una división por cero.

Cálculo del valor de paso por cada rayo

El valor de paso para los puntos de cada rayo se calcula mediante la **Ec. 3.26.**

$$step_i = step_{min} + \left(\frac{f(x_i) - f_{min}}{f_{max} + f_{min} + \varepsilon}\right) * (step_{max} - step_{min}) \tag{3.26}$$

donde $step_{max}\ y\ step_{min}$ son hiperparámetros constantes que limitan el valor de paso para los puntos en cada rayo. Los parámetros de la componente principal de la ecuación se encuentran descritas en la sección previa.

La idea detrás de tener número específico de puntos y valor de paso por cada rayo de acuerdo con su valor de aptitud es que cuando una "carga negativa" este cerca del optimo global genere mayor cantidad de puntos tanto para exploración como para explotación y que el valor de paso entre ellos sea menor, por el contrario, cuando la carga negativa está alejada del óptimo global, generará menor cantidad de puntos con una distancia mayor entre ellos.

Para simplificar el modelo, el número de puntos en las ramas secundarias es $SB_i = PB_i$.

Radio de desplazamiento

El radio de desplazamiento está definido con base en el valor de paso, el cual tiene una dirección tanto para la rama principal como para la secundaria.

En el caso de la rama principal, los puntos están orientados hacia el punto máximo global encontrado, y para la rama secundaria, la orientación es aleatoria generada por un número aleatorio entero en el rango $[-1, 1]$ de acuerdo con la dimensión de la función objetivo. Este proceso se muestra en los **Algoritmos 3.7 y 3.8.**

Generación de puntos por cada rayo

La posición de cada nuevo punto perteneciente al rayo i se obtiene mediante la **Ec. 3.27.**

$$NP_{p,d}^{i} = NP_{p-1,d}^{i} + rad_{p,d}^{i} \tag{3.27}$$

donde NP es el nuevo punto, ya sea de la rama principal o secundaria, que conforma el rayo i, p corresponde a la posición actual, d es el valor de la dimensión, $p-1$ la posición del punto anterior y rad el radio de desplazamiento.

Para dar una mayor robustez al algoritmo y para dar la forma característica de zigzag de los rayos reales, la posición del nuevo punto puede sufrir una *mutación*, en la cual se suma el valor calculado con un número aleatorio con distribución normal. Esto sucede cuando el valor de un número aleatorio superar la probabilidad de 0.5. Cuando esto sucede se aplica la **Ec. 3.28.**

$$NP_{p,d}^{i} = NP_{p,d}^{i} + rand_{N}(\mu, \sigma) \tag{3.28}$$

El **Algoritmo 3.7** resume la generación de las ramas principales en cada rayo.

Algoritmo 3.7: Generación de las ramas principales

Con $step$, $f_{max} = \min(f(x_i))$, $f_{min} = \max(f(x_i))$, $p_{min} = 5$, $p_{\max} = 15$ $\mu = 0$ and $\sigma = 0.2$, x_i es el conjunto de puntos iniciales.

- **for** i **in range (**n**):**
- $\mathrm{PB_i} = \mathrm{p_{min}} + \left(\frac{\mathrm{f(x_i)} - \mathrm{f_{min}}}{\mathrm{f_{max}} + \mathrm{f_{min}} + \varepsilon}\right) * (\mathrm{p_{max}} - \mathrm{p_{min}})$
- $step_i = step_{min} + \left(\frac{f(x_i) - f_{min}}{f_{max} + f_{min} + \varepsilon}\right) * (step_{max} - step_{min})$
- # ----------
- # **Dirección para los puntos de la rama principal**
- # ----------
- $\mathrm{dir} = \frac{\mathrm{Xbest}}{\mathrm{abs(Xbest)}}$
- # ----------
- # **Calculo de los nuevos puntos**
- # ----------
- **for** p **in range (**$\mathrm{PB_i}$**):**
  - $\mathrm{rad_p^i} = \mathrm{step} * \mathrm{dir}$
  - **if (**$p == 0$**):** $\mathrm{NP_p^i} = \mathrm{x_i}$
  - **else:** $\mathrm{NP_p^i} = \mathrm{NP_{p-1}^i} + \mathrm{rad_p^i}$
  - # ----------
  - # **Mutación**
  - # ----------
  - **if** $rand(0,1) > 0.5$:
    - $\mathrm{NP_{p,d}^i} = \mathrm{NP_{p,d}^i} + \mathrm{rand_N}(\mu, \sigma)$
- NP^i es el conjunto de puntos de la rama principal para el leader tip i.

Generación de ramas secundarias

Los puntos de la rama principal se evalúan de acuerdo con una regla de probabilidad para que los más aptos generen ramas secundarias. Esta selección está basada en la **Ec. 3.29.**

$$prob = exp\left(-\beta * \frac{f(NP_p^i)}{f_{max} + \varepsilon}\right) \tag{3.29}$$

donde β es una constante que controla la selección, $f(NP_p^i)$ es el valor de la función de aptitud para cada punto de la rama principal ($p = 1, 2, .., PB_i$) and f_{max} es el mejor valor global para la función de aptitud encontrada.

El **Algoritmo 3.8**. resume la generación de las ramas secundarias en cada rayo.

Algoritmo 3.8: Generación de ramas secundarias.

Con $\beta = 0.3$ y la rama principal NP^i.

- **for** i **in range (**n**):**
- $SB_i = PB_i$
-
- **# ----------**
- **# Regla de selección mediante ruleta**
- **# ----------**
- $prob = exp\left(-\beta * \frac{f(NP_p^i)}{f_{max}+\varepsilon}\right)$
- **if (**$prob > rand(0,1)$**):**
- **# ----------**
- **# Nuevos puntos para la rama secundaria**
- **# ----------**
- **for** p **in range (**SB_i**):**
- $dir = random.randint(-1, 1, D)$
- $rad_p^i = step * dir$
- **if (**$p == 0$**):** $SP_p^i = NP_p^i$
- **else:** $SP_p^i = SP_{p-1}^i + rad_p^i$
- **# ----------**
- **# Mutación**
- **# ----------**
- **if** $rand(0,1) > 0.5$:
- $SP_{p,d}^i = SP_{p,d}^i + rand_N(\mu, \sigma)$
-
- El rayo total para el punto inicial x_i es el conjunto L_i definido como $\{NP^i \cup SP^i\}$.

Mejores posiciones locales

Cuando el rayo de cada punto x_i ha sido formado completamente, se busca el punto que mejore el valor de la función de aptitud $f(x_i)$, a través de toda la trayectoria trazada. A este punto se lo denomina la mejor posición local.

Al inicio, las mejores posiciones P_i serán igual al conjunto de posiciones iniciales x_i y a lo largo de las iteraciones, estos puntos serán actualizados con aquellos con mejor valor en la función de aptitud. Este proceso se resume en el **Algoritmo 3.9.**

Algoritmo 3.9: Mejores posiciones locales.

Con $P_i = x_i$ definido antes del bucle de iteraciones y posterior a obtener los rayos L^i con el *Algoritmo 3.7*.

- **for** i **in range (**n**):**
- $mL_i = \min(L_i)$
- **if (**$f(mL_i) < f(P_i)$**):**
- $P_i = mL_i$

Actualización de posiciones

Las posiciones del conjunto de "cargas negativas" x_i se actualizan al final de cada iteración. La **Ec. 3.30 y Ec. 3.31** muestran la estrategia usada por el algoritmo LiA para calcular el desplazamiento de cada punto, con la finalidad de encontrar el óptimo global.

$$aux_i{}^1 = aux_i{}^1 + rand * (Xbest - P_i) \tag{3.30}$$

$$aux_i{}^2 = aux_i{}^2 + rand * (Xb - X_i) \tag{3.31}$$

donde i es el número de la partícula del conjunto $[1, 2, \dots, n]$, $Xbest$ es la ubicación del mejor punto encontrado y P es la mejor posición local del punto i y X es la posición del leader tip. El **Algoritmo 3.10** es usado para actualizar las posiciones del conjunto x_i. En este caso, es necesario definir previamente un conjunto de posiciones auxiliares $aux_i = x_i$ para prevenir la pérdida de la información encontrada.

Algoritmo 3.10: Actualización de posiciones.

- **for** i **in range (**n**):**
- $aux_i{}^1 = P_i$
- $aux_i{}^2 = X_i$
- $aux_i{}^1 = aux_i{}^1 + rand * (Xbest - P_i)$
- $aux_i{}^2 = aux_i{}^2 + rand * (Xbest - X_i)$
- **if (**$(aux_i{}^1) < f(P_i)$**):**
- $X_i = aux_i{}^1$
- **elif (**$(aux_i{}^2) < f(P_i)$**):**
- $X_i = aux_i{}^2$
- **else:**
- $X_i = P_i$
- x_i tiene una nueva posición al final de cada iteración.

Resumen del algoritmo propuesto LiA

El algoritmo propuesto está basado en una forma simple del modelo de un rayo real, con la finalidad de aplicarse en problemas de optimización. Las componentes principales son ramas: principal y secundarias generadas por cada punto en el espacio de soluciones.

Alrededor de $(n + RP + RS)$ funciones se evalúan durante cada iteración. Si el óptimo global es encontrado en la iteración T, entonces la complejidad de LiA puede ser descrita como: $O\big(T * (n + PB + SB)\big)$. En el **Algoritmo 3.11** se resume el algoritmo presentado.

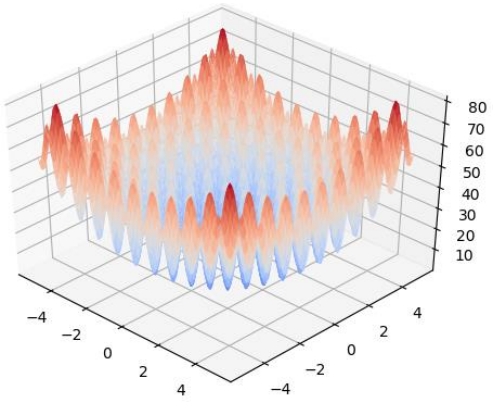
Algoritmo 3.11: Metaheurística de LiA.

- Inicializar los n puntos iniciales. **Algoritmo 3.6.**
- Inicializar las mejores posiciones locales con las posiciones iniciales.
- **for** itr **in range (**$iteraciones$**):**
- Encontrar f_{min} and f_{max}. (**Ec. 3.25**)
- **for** i **in range (**n**):**
- Calcular el número de puntos para la rama principal por cada posición i.(**Ec. 3.25**)
- Calcular el valor de paso para los puntos de cada rayo. (**Ec. 3.26**)
- Generar la rama principal por cada punto x_i. **Algoritmo 3.7.**
- Calcular el número de puntos para las ramas secundarias.
- Generar las ramas secundarias para cada punto x_i. **Algoritmo 3.8.**
- **for** i **in range (**n**):**
- Evaluar las mejores posiciones locales. **Algoritmo 3.9.**
- Actualizar las posiciones de cada punto x_i. **Algoritmo 3.10.**

Dinámica interna de LiA

La función de Rastrigin ha sido utilizada para mostrar la dinámica interna de LiA durante la búsqueda del óptimo global. Esta función está definida en 2 dimensiones con un tamaño de grilla de 100 por cada componente del valor de dimensión.

Los hiperparámetros usados por LiA son: un conjunto de 20 "cargas negativas" (*leader tip*), 8 puntos máximos y 4 puntos mínimos para cada rayo de acuerdo con su función de aptitud, el valor de paso mínimo es 2 y el valor máximo es 4. Se utilizaron 100 iteraciones. En la **Fig. 3.16**, se presenta la dinámica que ejecuta internamente LiA.



Función de Rastrigin en perspectiva 3D

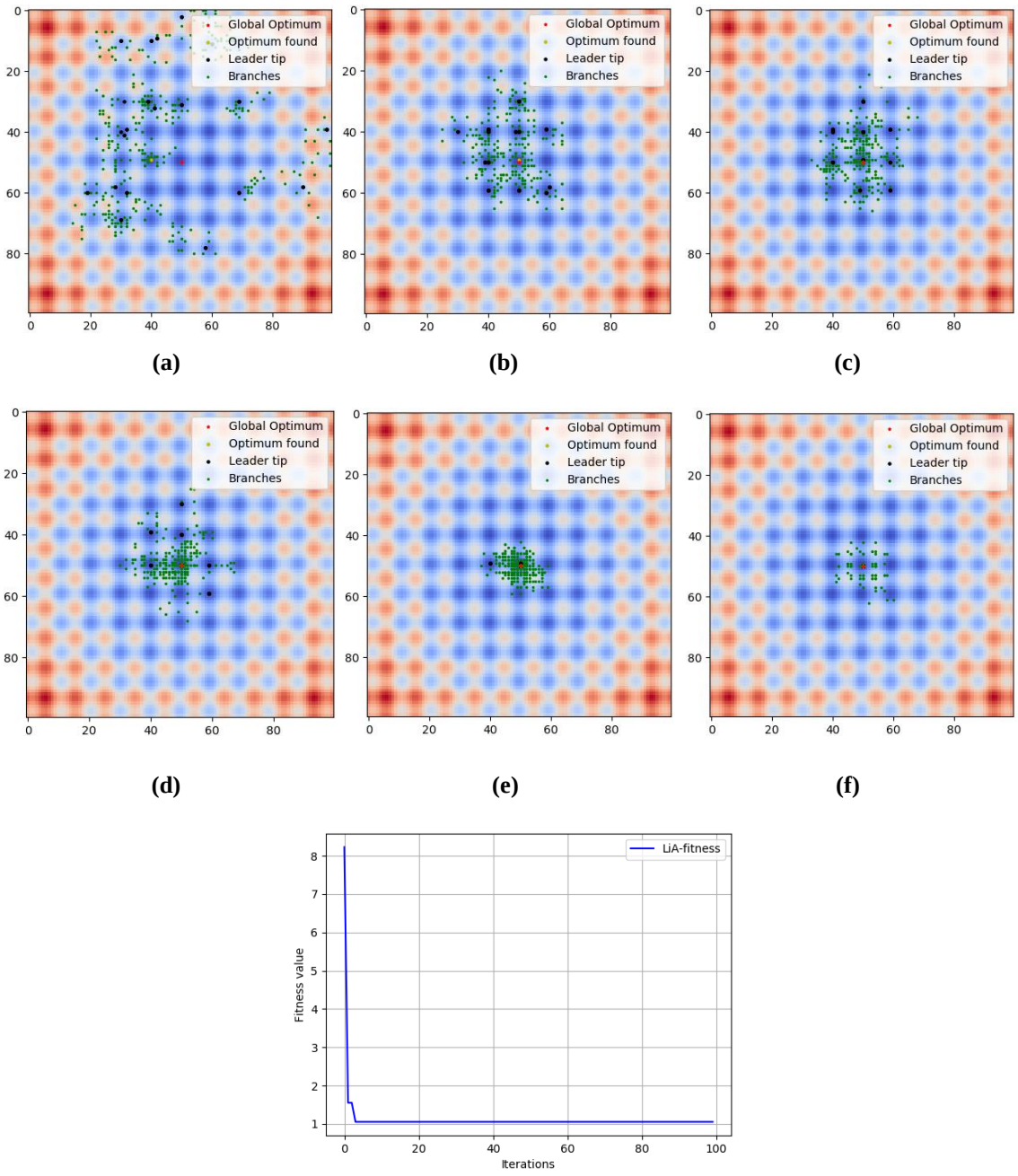


Fig. 3.16: Dinámica interna de LiA **(a)** Iteración 1 de LiA. El punto rojo es el óptimo global, los puntos negros es el conjunto de cargas iniciales, los puntos verdes son los puntos de cada rayo. **(b)** Iteración 10. **(c)** Iteración 20. **(d)** Iteración 30. **(e)** Iteración 50. **(f)** Iteración 100. **(g)** Curva de convergencia de LiA.

Banco de pruebas iniciales

Con la finalidad de analizar el rendimiento del algoritmo propuesto, se lo ha comparado con los algoritmos PSO, FWA y ABC, por medio de 12 funciones de pruebas utilizadas por varios algoritmos de manera estándar. Estas funciones de prueba se muestran en la **Tabla 3.5.**

Tabla 3.5: Funciones de prueba utilizadas por varios algoritmos de inteligencia colectiva para analizar su rendimiento.

Función	Expresión matemática	Rango de inicialización	Dimensión
Ackley	$F_1 = -20\,exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - exp\left(\frac{1}{D}\sum_{i=1}^{D} cos\,(2\pi x_i^2)\right) + \varepsilon$	$[-5.12, 5.12]^D$	2
Sphere	$F_2 = \sum_{i=1}^{D} x_i^2$	$[-100, 100]^D$	3
Rosenbrock	$F_3 = \sum_{i=1}^{D-1}(100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2)$	$[-5.0, 5.0]^D$	2
Rastrigin	$F_4 = \sum_{i=1}^{D}(x_i^2 - 10(cos\,(2\pi x_i^2)) + 10)$	$[-5.12, 5.12]^D$	3
Cigar	$F_5 = x_1^2 + \sum_{i=2}^{D} 10^4\, x_i^2$	$[-5.0, 5.0]^D$	2
Griewank	$F_6 = 1 + \sum_{i=1}^{D} \frac{x_i}{4000} - \prod_{i=1}^{D} cos\,\left(\frac{x_i}{\sqrt{i}}\right)$	$[-600, 600]^D$	3
Schwefel	$F_7 = \sum_{i=1}^{D}((x_1 - x_i^2)^2 + (x_i - 1)^2)$	$[-500, 500]^D$	2
Drop-wave	$F_8 = \frac{1 + cos\,\left(12\sqrt{\sum_{i=1}^{D} x_i^2}\right)}{0.5(\sum_{i=1}^{D} x_i^2) + 2}$	$[-5.12, 5.12]^D$	3
Levy	$F_9 = sin^2(\pi\omega_1) + \sum_{i=1}^{D-1}(\omega_i - 1)^2[1 + 10sin^2(\pi\omega_i + 1)] + (\omega_d - 1)^2[1 + sin^2(2\pi\omega_d)]$ $where\ \ \omega_i = 1 + \frac{x_i + 1}{4}$	$[-10, 10]^D$	2
Langermann	$F_{10} = \sum_{i=1}^{5} c_i\, exp\left(-\frac{1}{\pi}\sum_{j=1}^{D}(x_j + A_{ij})^2\right) cos\left(\pi\sum_{j=1}^{D}(x_j + A_{ij})^2\right)$	$[0, 10]^D$	2
Himmelblau	$F_{11} = (x^2 + y - 11)^2 + (y^2 + x - 7)^2$	$[-5, 5]^D$	2
Beale	$F_{12} = (1.5 - x + xy)^2 + (2.25 - x + xy^2)^2 + (2.625 - x + xy^3)^2$	$[-4.5, 4.5]^D$	2

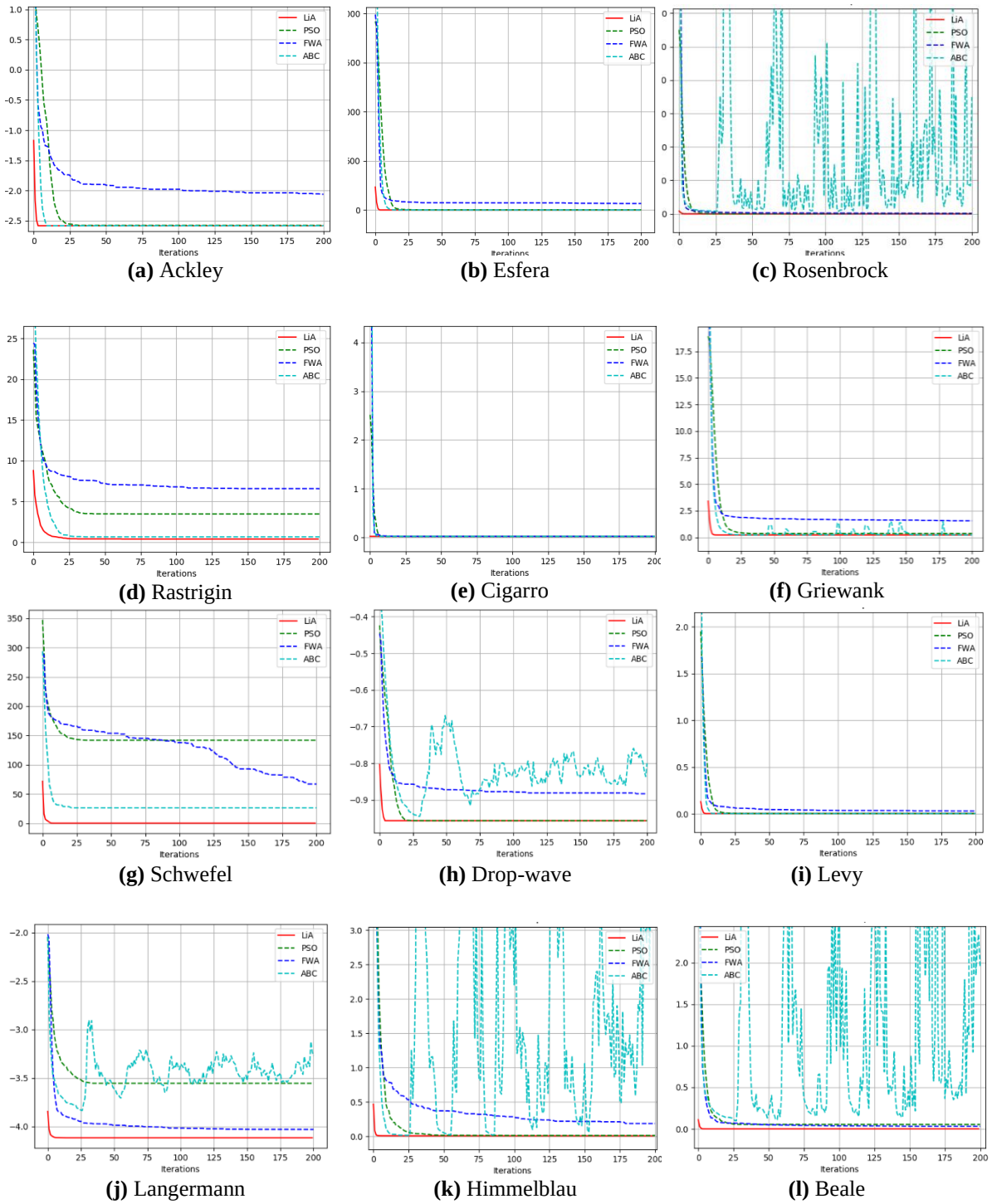
Resultados obtenidos

Para las pruebas se utilizó una población de 20 partículas por cada algoritmo, 200 iteraciones y 50 ejecuciones independientes para analizar su rendimiento y estabilidad durante el proceso de optimización. En la **Tabla 3.6**, se muestran los valores estadísticos como el promedio y la desviación estándar obtenidos con cada algoritmo.

Tabla 3.6: Promedio y desviación estándar obtenidos con cada algoritmo durante las pruebas.

Función	***LiA*** ***Mean - STD***	***PSO*** ***Mean - STD***	***FWA*** ***Mean - STD***	***ABC*** ***Mean - STD***
Ackley	-2.580 4.44×10^{-16}	-2.575 0.036	-2.059 0.430	-2.580 4.44×10^{-16}
Sphere	0.758 1.11×10^{-16}	1.889 1.409	66.655 46.440	0.757 1.11×10^{-16}
Rosenbrock	0.0025 4.34×10^{-19}	0.026 0.013	0.121 0.107	34.965 139.38
Rastrigin	0.393 0.0	3.465 1.850	6.556 2.319	0.657 0.402
Cigar	252519.13 0.0	252519.13 0.0	252546.80 51.97	252519.13 0.0
Griewank	0.222 2.78×10^{-17}	0.367 0.119	1.556 0.459	0.222 1.33×10^{-15}
Schwefel	0.474 0.0	141.950 92.349	67.255 48.70	26.588 59.600
Drop-wave	-0.956 1.11×10^{-16}	-0.956 0.0	-0.883 0.068	-0.799 0.222
Levy	0.003 0.0	0.004 0.0	0.031 0.029	0.003 0.0
Langermann	-4.116 8.88×10^{-16}	-3.554 0.852	-4.029 0.054	-3.271 0.845
Himmelblau	0.0078 0.0009	0.0132 0.010	0.1866 0.182	3.29 9.73
Beale	0.0015 2.17×10^{-19}	0.053 0.205	0.03 0.028	1.96 5.463

En la **Fig. 3.17**, se muestra el promedio de los valores en las 50 ejecuciones, durante la minimización cada función objetivo. En esta figura se puede ver además la comparación del rendimiento entre los algoritmos utilizados.



(a) Ackley **(b)** Esfera **(c)** Rosenbrock

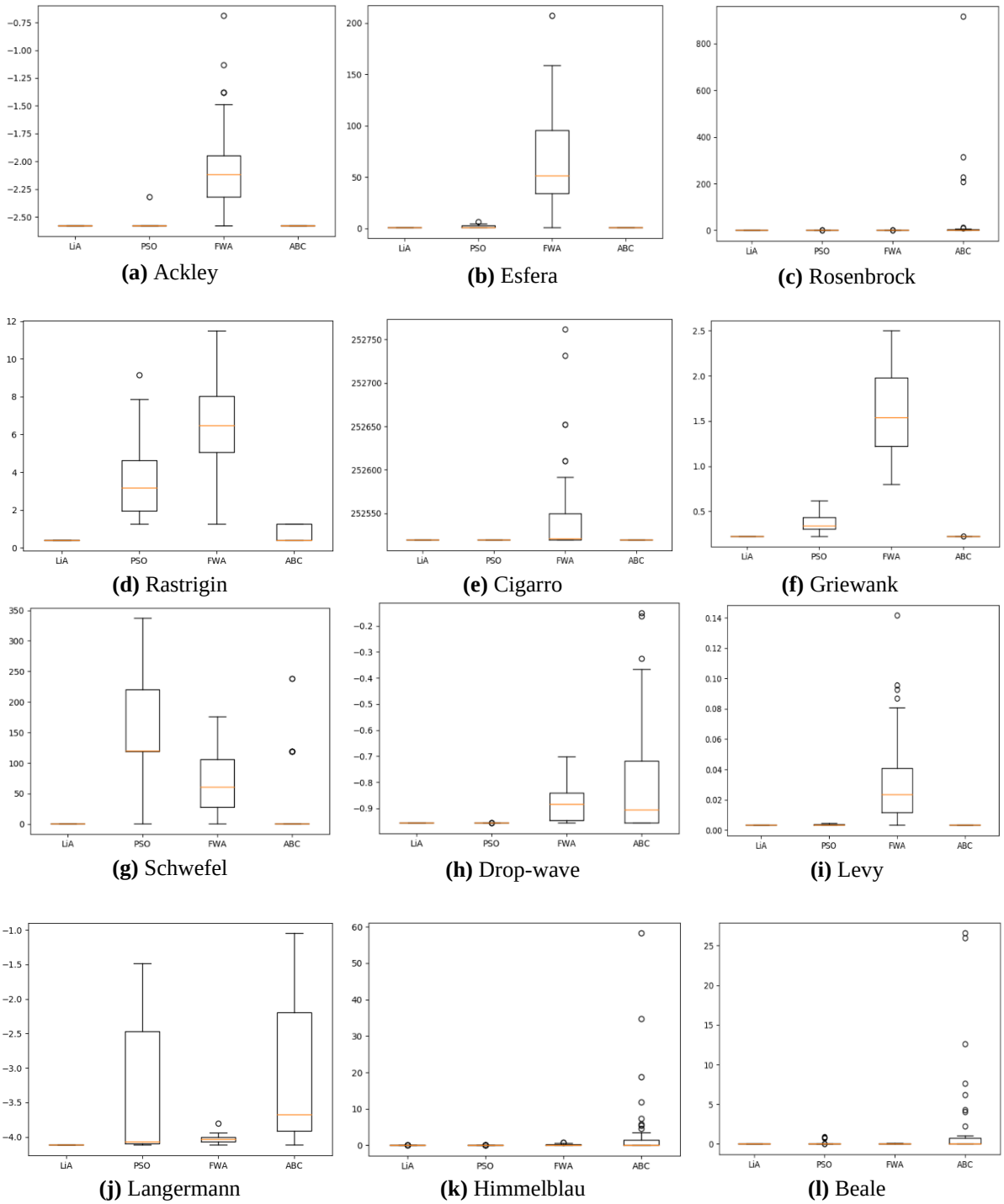
(d) Rastrigin **(e)** Cigarro **(f)** Griewank

(g) Schwefel **(h)** Drop-wave **(i)** Levy

(j) Langermann **(k)** Himmelblau **(l)** Beale

Fig. 3.17: Comparación de las curvas de convergencias en cada función de prueba, alcanzadas por los algoritmos utilizados en la comparación.

En la **Fig. 3.18**, se muestra el análisis de la distribución estadística de los resultados obtenidos por cada algoritmo. Estos diagramas de caja (*boxplots*) permiten saber la estabilidad de la convergencia de cada algoritmo.



(a) Ackley **(b)** Esfera **(c)** Rosenbrock

(d) Rastrigin **(e)** Cigarro **(f)** Griewank

(g) Schwefel **(h)** Drop-wave **(i)** Levy

(j) Langermann **(k)** Himmelblau **(l)** Beale

Fig. 3.18: Análisis de la distribución de los resultados obtenidos para comparación de la estabilidad en la convergencia de cada algoritmo.

Conclusiones a partir del análisis de los resultados

LiA tiene un balance adecuado para el criterio de exploración y explotación, lo que permite mejorar la búsqueda del óptimo global en el espacio de la función objetivo. Teniendo en cuenta que la inicialización es siempre un paso crucial en los algoritmos estocásticos, la inicialización caótica se ha utilizado para mejorar el rendimiento de LiA.

Se ha realizado las pruebas con diez funciones en 2 y 3 dimensiones, que muestran resultados sólidos en términos de velocidad de convergencia, la cual es mejor en varios casos a los obtenidos

por los algoritmos PSO, FWA y ABC; sin embargo, en términos de estabilidad y precisión de optimización, LiA supera ampliamente en todos los casos a los obtenidos por PSO, FWA y ABC. Además, se utilizó la librería estándar COCO para analizar su rendimiento en 5, 10 y 20 dimensiones, la cual, es usada en la evaluación de nuevos algoritmos de SI.

Comparación del algoritmo LiA aplicado en registración rígida

El aporte de esta sección es la evaluación del rendimiento del algoritmo propuesto LiA, aplicado en registración rígida. Además, los resultados del rendimiento se comparan con los algoritmos de optimización utilizados en la sección 3.7.2.

Datos de las pruebas

Para las pruebas se realizaron 20 ejecuciones independientes de cada algoritmo, el número de iteraciones es 100 y el número de partículas es 30. Para la comparación del rendimiento de los algoritmos se utilizaron las imágenes de la **Fig. 3.10,** para las dos modalidades de registración.

Registración monomodal

La registración monomodal se realiza entre imágenes de la misma modalidad T1-T1 y T2-T2. La **Fig. 3.19** muestra el promedio de la *correlación de Pearson* alcanzado por cada algoritmo durante las iteraciones por cada prueba, además, se muestra la gráfica de la distribución de los datos de los resultados obtenidos.

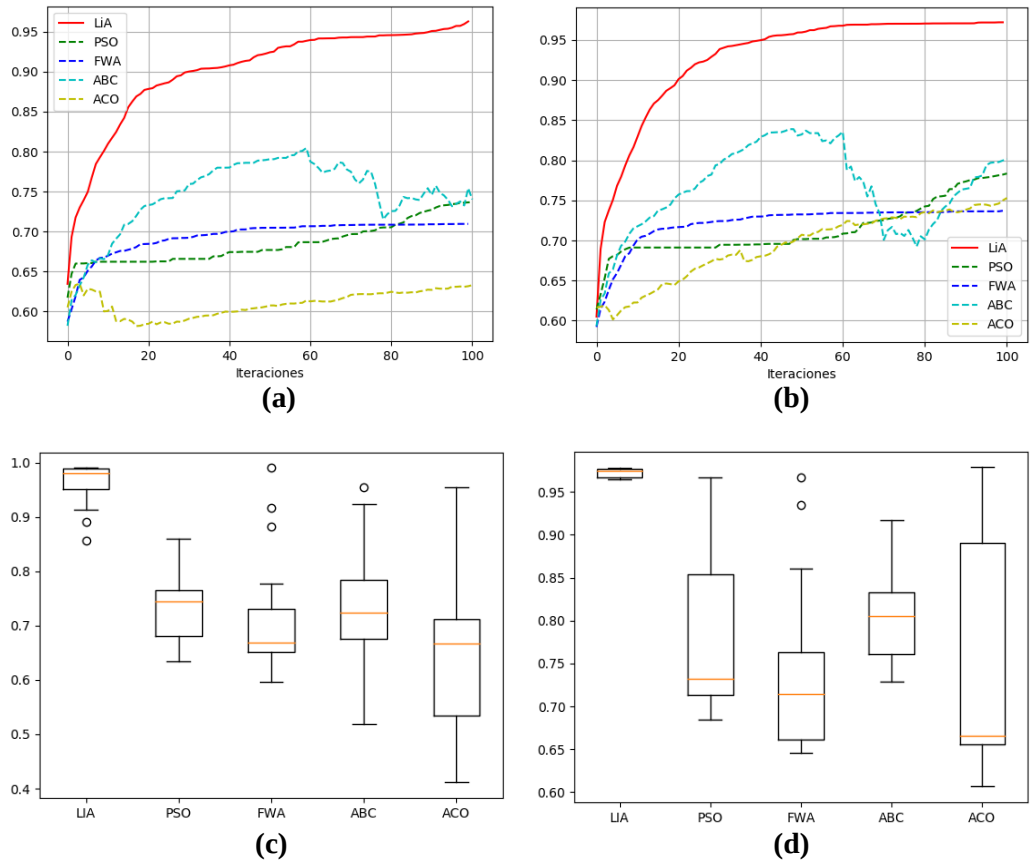


Fig. 3.19: Gráficas del promedio de la correlación de Pearson por cada algoritmo. **(a) Prueba 1** entre las imágenes 1 y 2 superiores de Fig. 3.10 **(b) Prueba 2** entre las imágenes inferiores 1 y 2 de Fig. 3.10. **(c)** Gráfica de la distribución estadística de los resultados de la prueba 1. **(d)** Gráfica de la distribución estadística de los resultados de la prueba 2.

En las **Tabla 3.7 y 3.8**, se muestran los valores más representativos obtenidos por cada algoritmo en las pruebas monomodales 1 y 2.

Tabla 3.7: Valores de correlación de Pearson obtenidos en la prueba 1.

Algoritmo	**Promedio *valor de correlación***	**Desviación estándar**	**Max. *valor de correlación***
LiA	0.96	0.037	0.99
PSO	0.74	0.064	0.86
FWA	0.71	0.103	0.98
ABC	0.74	0.108	0.96
ACO	0.63	0.125	0.95

Tabla 3.8: Valores de correlación de Pearson obtenidos en la prueba 2.

Algoritmo	**Promedio *valor de correlación***	**Desviación estándar**	**Max. *valor de correlación***
LiA	0.97	0.005	0.98
PSO	0.78	0.094	0.96
FWA	0.74	0.092	0.97
ABC	0.80	0.051	0.92
ACO	0.75	0.135	0.98

Registración multimodal

La registración multimodal se realiza entre imágenes de distinta modalidad T1-T2 y T2-SPECT. La **Fig. 3.20** muestra el promedio de la *información mutua* alcanzado por cada algoritmo durante las iteraciones por cada prueba, además, se muestra la gráfica de la distribución de los datos de los resultados obtenidos.

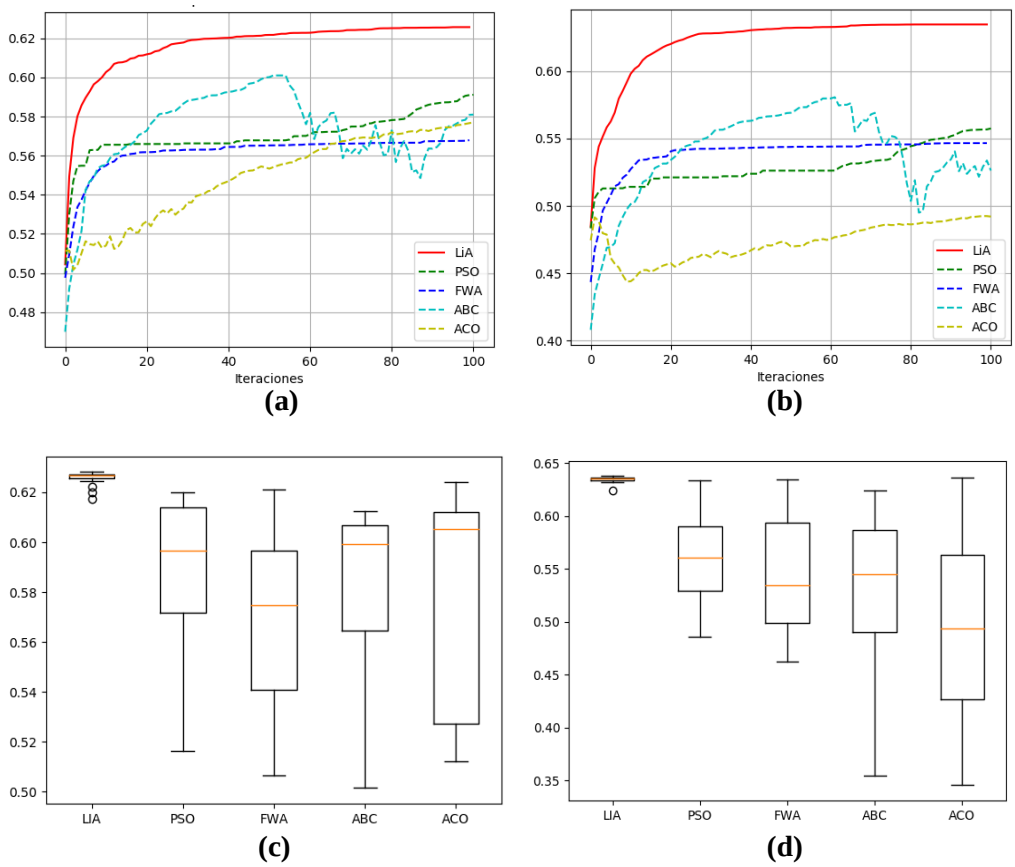


Fig. 3.20: Gráficas del promedio de la información mutua por cada algoritmo. **(a) Prueba 3** entre las imágenes 1 y 3 superiores de Fig. 3.10 **(b) Prueba 4** entre las imágenes inferiores 1 y 3 de Fig. 3.10. **(c)** Gráfica de la distribución estadística de los resultados de la prueba 3. **(d)** Gráfica de la distribución estadística de los resultados de la prueba 4.

En las **Tabla 3.9 y 3.10**, se muestran los valores más representativos obtenidos por cada algoritmo en las pruebas multimodales 3 y 4.

Tabla 3.9: Valores de información mutua obtenidos en la prueba 3.

Algoritmo	**Promedio *valor de MI***	**Desviación estándar**	**Max. *valor de MI***
LiA	0.63	0.003	0.63
PSO	0.59	0.026	0.62
FWA	0.57	0.036	0.62
ABC	0.58	0.036	0.61
ACO	0.58	0.044	0.62

Tabla 3.10: Valores de información mutua obtenidos en la prueba 4.

Algoritmo	**Promedio *valor de MI***	**Desviación estándar**	**Max. *valor de MI***
LiA	0.63	0.003	0.64
PSO	0.56	0.041	0.63
FWA	0.55	0.058	0.63
ABC	0.53	0.074	0.62
ACO	0.49	0.087	0.64

Conclusiones a partir del análisis de los resultados

De acuerdo con los resultados obtenidos, LiA es el algoritmo que mejor rendimiento presenta para la registración monomodal. El segundo mejor algoritmo es ABC (Optimización por colonia de abejas) basado en [66], frente al problema de registración rígida monomodal. Los valores representativos de las **Tablas 3.7 y 3.8** corroboran esta conclusión donde se observa que presenta un factor promedio de correlación superior al 0.95 en las dos pruebas.

En el caso de la registración multimodal, LiA es igualmente el que mejor rendimiento presenta con un valor promedio superior al 0.60 del factor de Información Mutua para las dos pruebas. El segundo mejor es ABC. Los datos representativos mostrados en las **Tablas 3.9 y 3.10** corroboran esta conclusión.

Además, LiA es el mejor algoritmo, tanto para registración monomodal como multimodal, de acuerdo con la distribución estadística de los datos de salida, ya que presenta una mínima desviación estándar comparada con las obtenidas por los demás algoritmos, por tanto, la probabilidad de encontrar un buen resultado en una ejecución es alta.

Referencias

[1] E. K. P. Chong and S. H. Zak, *An Introduction to Optimization*, 2nd. Wiley, 2001.

[2] D. Luenberger and Y. Ye, *Linear and Nonlinear Programming*, Third. 2008.

[3] S. Boyd and L. Vandenberghe, *Convex Optimization*, First. Cambridge University Press, 2004.

[4] C. Bishop, *Pattern Recognition and Machine Learning*. 2006.

[5] A. Swietanowski, “SIMPLEX v. 2.17: an Implementation of the Simplex Algorithm for Large Scale Linear Problems.,” Laxenburg - Austria, 1994.

[6] A. Alush and J. Goldberger, “Ensemble Segmentation Using Efficient Integer Linear Programming,” *IEEE PATTERN Recognit. Mach. Intell.*, no. 1, 2012.

[7] K. Tsuda and G. Ratsch, “Image reconstruction by linear programming,” *IEEE Trans. Image Process.*, vol. 14, no. 6, pp. 737–744, Jun. 2005.

[8] W. Sun and Y.-X. Yuan, *Optimization Theory And Methods Nonlinear Programming*, First. Springer, 2006.

[9] L. Huang, W. Zhao, B. R. Abidi, and M. A. Abidi, "A Constrained Optimization Approach for Image Gradient Enhancement," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 8, pp. 1707–1718, Aug. 2018.

[10] M. Sdika, "A Fast Nonrigid Image Registration With Constraints on the Jacobian Using Large Scale Constrained Optimization," *IEEE Trans. Med. Imaging*, vol. 27, no. 2, pp. 271–281, Feb. 2008.

[11] M. V Afonso, J. M. Bioucas-Dias, and M. A. T. Figueiredo, "An Augmented Lagrangian Approach to the Constrained Optimization Formulation of Imaging Inverse Problems," *IEEE Trans. IMAGE Process.*, vol. 20, no. 3, 2011.

[12] D. Oliva and E. Cuevas, "Optimization," in *Advances and Applications of Optimised Algorithms in Image Processing*, Springer, 2017, pp. 13–21.

[13] S. Ruder, "An overview of gradient descent optimization algorithms *," Dublin, 2016.

[14] E. Vural and P. Frossard, "Analysis of Descent-Based Image Registration," *SIAM J. Imaging Sci.*, vol. 6, no. 4, pp. 2310–2349, 2013.

[15] V. M. Patel, R. Maleh, A. C. Gilbert, and R. Chellappa, "Gradient-Based Image Recovery Methods From Incomplete Fourier Measurements," *IEEE Trans. IMAGE Process.*, vol. 21, no. 1, 2012.

[16] M. Mardani *et al.*, "Neural Proximal Gradient Descent for Compressive Imaging," *32nd Conf. Neural Inf. Process. Syst. (NIPS 2018)*, 2018.

[17] J. W. Chinneck, "Introduction to Nonlinear Programming," in *Practical Optimization: a gentle Introduction*, 2015.

[18] J. Mu and D. Z. Chen, "An optimization-based approach for restoring missing structures and textures in images," in *2015 IEEE International Conference on Image Processing (ICIP)*, 2015, pp. 3705–3709.

[19] I. Bashir, A. Majeed, and O. Khursheed, "Image restoration and the various restoration techniques used in the field of Digital Image processing," *Int. J. Comput. Sci. Mob. Comput.*, vol. 6, no. 6, pp. 390–393, 2017.

[20] M. Cavazzuti, "Deterministic Optimization," in *Optimization Methods*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 77–102.

[21] M.-H. Lin, J.-F. Tsai, and C.-S. Yu, "A Review of Deterministic Optimization Methods in Engineering and Management," *Math. Probl. Eng.*, vol. 2012, pp. 1–15, 2012.

[22] M. Alejandro Pereyra *et al.*, "Stochastic Simulation and Optimization Methods in Signal Processing," *IEEE J. Sel. Top. Signal Process.*, vol. 10, no. 2, pp. 224–241, 2015.

[23] L. A. Hannah, "Stochastic Optimization," *Int. Encycl. Soc. Behav. Sci.*, pp. 473–481, 2015.

[24] P. Collet and J.-P. Rennard, "Stochastic Optimization Algorithms," *Rennard, J.-P., Handb. Res. Nat. Inspired Comput. Econ. Manag.*, 2007.

[25] A. Ruszczyński and A. Shapiro, "Stochastic Programming Models," *Handbooks Oper. Res. Manag. Sci.*, vol. 10, pp. 1–64, Jan. 2003.

[26] A. Stefek, "Benchmarking of heuristic optimization methods," in *14th International Conference Mechatronika*, 2011, pp. 68–71.

[27] M. Dorigo and T. Stützle, *Ant colony optimization*. MIT Press, 2004.

[28] G. B. Thomas, M. D. Weir, and J. R. Hass, *Thomas' calculus : early transcendentals.*, 12th ed. Addison-Wesley, 2010.

[29] J. Xu and J. Zhang, "Exploration-exploitation tradeoffs in metaheuristics: Survey and analysis," in *Proceedings of the 33rd Chinese Control Conference*, 2014, pp. 8633–8638.

[30] A. Simpkins, R. de Callafon, and E. Todorov, "Optimal trade-off between exploration and exploitation," in *2008 American Control Conference*, 2008, pp. 33–38.

[31] M. Tokic and G. Palm, "Gradient Algorithms for Exploration/Exploitation Trade-Offs: Global and Local Variants," Springer, Berlin, Heidelberg, 2012, pp. 60–71.

[32] J. Hoffmeyer, "The Swarming Body," *Semiot. Around World. Proc. Fifth Congr. Int. Assoc. Semiot. Stud.*, pp. 937–940, 1994.

[33] A. P. Engelbrecht, "Computational intelligence: An introduction," in *Studies in Computational Intelligence*, 2nd ed., 2007.

[34] M. Dorigo, E. Bonabeau, and G. Theraulaz, "Ant algorithms and stigmergy," 2000.

[35] C. Blum and X. Li, "Swarm Intelligence in Optimization," in *Swarm Intelligence*, Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 43–85.

[36] C. P. Lim and L. C. Jain, "Advances in Swarm Intelligence," Springer, Berlin, Heidelberg, 2009, pp. 1–7.

[37] S. Roy, S. Biswas, and S. Sinha Chaudhuri, "Nature-Inspired Swarm Intelligence and Its Applications," *Int. J. Mod. Educ. Comput. Sci.*, vol. 6, no. 12, pp. 55–65, Dec. 2014.

[38] S. Das, A. Abraham, and A. Konar, "Swarm Intelligence Algorithms in Bioinformatics," Springer, Berlin, Heidelberg, 2008, pp. 113–147.

[39] E. Bonabeau and C. Meyer, "Swarm intelligence. A whole new way to think about business.," *Harv. Bus. Rev.*, vol. 79, no. 5, pp. 106–14, 165, May 2001.

[40] M. M. Millonas, "Swarms, Phase Transitions, and Collective Intelligence," 1993.

[41] M. Dorigo, M. Birattari, and T. Stützle, "Ant Colony Optimization – Artificial Ants as a Computational Intelligence Technique," *IEEE Comput. INTELL. MAG*, vol. 1, pp. 28-39, 2006.

[42] J. Huo, Z. Wang, F. T. S. Chan, C. K. M. Lee, and J. O. Strandhagen, "Assembly Line Balancing Based on Beam Ant Colony Optimisation," *Math. Probl. Eng.*, vol. 2018, pp. 1–17, Oct. 2018.

[43] J. Yang, X. Shi, M. Marchese, and Y. Liang, "An ant colony optimization method for generalized TSP problem," *Prog. Nat. Sci.*, vol. 18, no. 11, pp. 1417–1422, Nov. 2008.

[44] T. B. Kurniawan, Z. Ibrahim, N. K. Khalid, and M. Khalid, "A Population-Based Ant Colony Optimization Approach for DNA Sequence Optimization," in *2009 Third Asia International Conference on Modelling & Simulation*, 2009, pp. 246–251.

[45] K. O. Jones and A. Bouffet, "Comparison of ant colony optimisation and differential evolution," in *Proceedings of the 2007 international conference on Computer systems and technologies - CompSysTech '07*, 2007, p. 1.

[46] H. Ahmed and J. Glasgow, "Swarm Intelligence: Concepts, Models and Applications," Ontario, Canada, 2012.

[47] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," *IEEE*, pp. 1942–1948, 1995.

[48] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, pp. 39–43.

[49] M. G. H. Omran, A. P. Engelbrecht, and A. Salman, "Particle Swarm Optimization for Pattern Recognition and Image Processing," 2006, pp. 125–151.

[50] R. C. Eberhart and Yuhui Shi, "Tracking and optimizing dynamic systems with particle swarms," in *Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No.01TH8546)*, vol. 1, pp. 94–100.

[51] S. GARG, K. PATRA, and S. K. PAL, "Particle swarm optimization of a neural network model in a machining process," *Sadhana*, vol. 39, no. 3, pp. 533–548, Jun. 2014.

[52] Q. Li and I. Sato, "Multimodality Image Registration by Particle Swarm Optimization of Mutual Information," *LNAI*, vol. 4682, pp. 1120–1130, 2007.

[53] M. P. Wachowiak, R. Smolikova, Y. Zheng, J. M. Zurada, and A. S. Elmaghraby, "An Approach to Multimodal Biomedical Image Registration Utilizing Particle Swarm Optimization," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 289–301, Jun. 2004.

[54] L. Messerschmidt and A. P. Engelbrecht, "Learning to Play Games Using a PSO-Based Competitive Learning Approach," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 280–288, Jun. 2004.

[55] T. M. Blackwell and P. Bentley, "Improvised music with swarms," in *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600)*, vol. 2, pp. 1462–1467.

[56] Y. Shi and R. C. Eberhart, "Empirical study of particle swarm optimization," in *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, pp. 1945–1950.

[57] M. Imran, R. Hashim, and N. E. A. Khalid, "An Overview of Particle Swarm Optimization Variants," *Procedia Eng.*, vol. 53, pp. 491–496, Jan. 2013.

[58] S. D. Chavan and N. P. Adgokar, "An Overview on Particle Swarm Optimization: Basic Concepts and Modified Variants," 2013.

[59] D. Karaboga and B. Basturk, "Artificial Bee Colony (ABC) Optimization Algorithm for Solving Constrained Optimization Problems," *Springer-Verlag Berlin Heidelb.*, vol. IFSA 2007, no. LNAI 4529, pp. 789–798, 2007.

[60] D. Karaboga and B. Basturk, "On the performance of artificial bee colony (ABC) algorithm," *Appl. Soft Comput.*, vol. 8, pp. 687–697, 2008.

[61] D. Teodorović, "Bee Colony Optimization (BCO)," in *Innovations in Swarm Intelligence*, 2009, pp. 39–60.

[62] J. A. Bullinaria and K. AlYahya, "Artificial Bee Colony Training of Neural Networks," Springer, Cham, 2014, pp. 191–201.

[63] D. Karaboga and C. Ozturk, "Neural networks training by artificial bee colony algorithm on pattern classification," *Neural Netw. World* , vol. 19, no. 3, pp. 279–292, 2009.

[64] C. Xu and H. Duan, "Artificial bee colony (ABC) optimized edge potential function (EPF) approach to target recognition for low-altitude aircraft," *Pattern Recognit. Lett.*, vol. 31, no. 13, pp. 1759–1772, Oct. 2010.

[65] P. K. Sharma, B. V S, N. K. M, S. K. S, and P. P, "Artificial Bee Colony and Its Application for Image Fusion," *Int. J. Inf. Technol. Comput. Sci.*, vol. 4, no. 11, pp. 42–49, Oct. 2012.

[66] W. Gao, S. Liu, and L. Huang, "A global best artificial bee colony algorithm for global optimization," *J. Comput. Appl. Math.*, vol. 236, no. 11, pp. 2741–2753, May 2012.

[67] R. Kumar, "Directed Bee Colony Optimization Algorithm," *Swarm Evol. Comput.*, vol. 17, pp. 60–73, Aug. 2014.

[68] Y. Tan and Y. Zhu, "Fireworks Algorithm for Optimization," *ICSI 2010 Adv. Swarm Intell.*, pp. 355–364, 2010.

[69] Y. Tan, "Fireworks Algorithm (FWA)," in *Fireworks Algorithm*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 17–35.

[70] S. Zheng, A. Janecek, and Y. Tan, "Enhanced Fireworks Algorithm," in *2013 IEEE Congress on Evolutionary Computation*, 2013, pp. 2069–2077.

[71] M. Guendouz, A. Amine, and R. M. Hamou, "A discrete modified fireworks algorithm for community detection in complex networks," *Appl. Intell.*, vol. 46, no. 2, pp. 373–385, Mar. 2017.

[72] R. Isa-Jara, F. J. Buchelly, G. J. Meschino, and B. V.L., "Improved Particle Swarm Optimization algorithm applied to rigid registration in medical images," Springer, Singapore, 2017, pp. 161–164.

[73] B. Clay Graham-Jones, "The Fractal Nature of Lightning: An Investigation of the Fractal Relationship of the Structure of Lightning to Terrain," Florida State University, 2006.

[74] V. A. Rakov, M. A. Uman, and M. A. Uman Frontmatter, *Lightning Physics and Effects-Lightning: Physics and Effects*, First. 2003.

Capítulo 4 Extracción de características

> La cosa más hermosa que podemos experimentar es el misterio. Es la fuente de todo arte y toda ciencia.
>
> *Albert Einstein*

4.1 Introducción

El sistema visual humano es reconocido como uno de los sistemas visuales biológicos más poderosos de la naturaleza. Esto radica principalmente en la forma que en capta la información del entorno, ya que lo puede hacer tanto de forma pasiva como activa. Desde las civilizaciones antiguas hasta la actualidad, se ha perseguido el sueño de construir máquinas que emulen el comportamiento humano inteligente con distintos propósitos, donde el sistema visual no ha sido la excepción.

Actualmente, el área dedicada a este propósito es la *Visión por computadora*[7], la cual reúne varios paradigmas con la finalidad de darle a las máquinas la capacidad de "ver y analizar" su entorno (extraer "conocimiento"), de modo que puedan extraer apariencias y estructuras que les permitan tomar decisiones. Esto se logra mediante una *representación visual computarizada* adecuada, la cual, aprovecha las técnicas de procesamiento informático disponibles.

Las tareas de reconocimiento, por ejemplo, permiten detectar objetos, reconocer escenas y categorizarlas de acuerdo con su contenido [1]. Para esto, se extraen de las imágenes o videos representaciones de información discriminativa conocidas como *características visuales o features*. Estas características pueden ser de tipo local o global y se utilizan para la construcción de diccionarios visuales. Actualmente se ha demostrado que las características locales son más robustas frente a las globales, especialmente, cuando se analizan imágenes captadas con variaciones en ángulos de visión, iluminación, cambios de escala, etc. La idea de extraer y describir características locales está inspirada en el estudio del seguimiento y análisis del movimiento [2].

4.2 Características locales

Las características locales permiten encontrar estructuras dentro de una imagen de manera repetible, es decir, que los resultados sean robustos frente a determinados cambios como la iluminación o contraste. Estas estructuras se codifican en una *representación* para establecer su invariancia frente a diversas transformaciones geométricas, tales como las isometrías, similaridad, afines o proyectivas. El propósito principal de estas características es que permitan realizar un emparejamiento eficiente de las estructuras locales entre dos o más imágenes [1].

[7] En inglés *Computer Vision*, es un área de investigación que vincula varias disciplinas, tales como: el procesamiento de imágenes, la inteligencia computacional, el reconocimiento de patrones, la probabilidad y la estadística, entre otras.

4.2.1 Definición de una característica local

Una característica local es una fuente de información considerada como una medida de la imagen con relevancia estadística, algebraica, geométrica, espacial o espectral. Ésta puede ser un punto, una esquina, un borde o una región pequeña de un objeto o de una zona dentro de la imagen. Se la considera un patrón, puesto que, se diferencia notablemente de su vecindario inmediato. Generalmente están asociadas con un cambio de una o más propiedades de una imagen, como la intensidad, el color y la textura, aunque no siempre están localizadas sobre ese cambio [3].

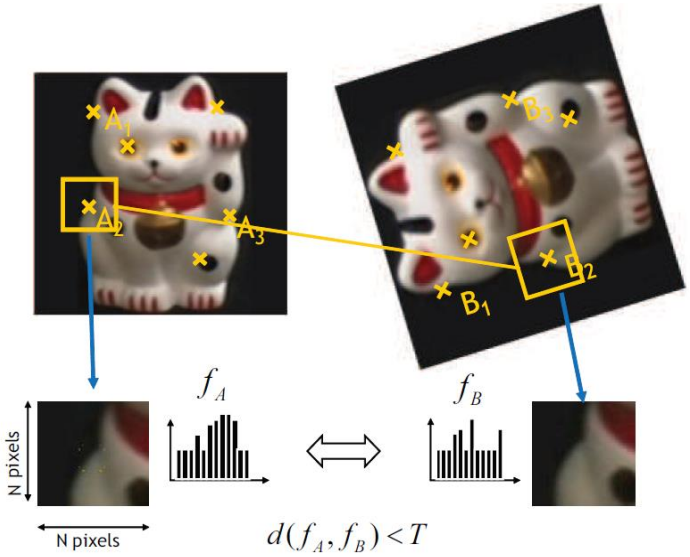


Fig. 4.1: Identificación de características locales para reconocimiento entre dos imágenes. Las características locales están asociadas a una región de la imagen, donde ella es el centro de dicha región. **Fuente:** K. Grauman et al., Visual Object Recognition. Cap. 3, pag. 12, 2010.

En la **Fig. 4.1**, se muestra el reconocimiento de objetos entre dos imágenes, cuando una de ellas ha sido deformada mediante una transformación de similaridad i.e., cambio en la escala, cambio en la orientación y/o cambio en la posición. A modo de ejemplo se muestra el emparejamiento de la característica **A2** con la característica **B2**. Cada una de ellas tiene una extensión especial, es decir, un vecindario local de píxeles donde el centro de dicha región se utiliza como identificador. Para el emparejamiento, se establece una métrica de distancia y un valor umbral T, con la condición de que la distancia sea menor que T para que la correspondencia sea aceptada.

Este proceso se diferencia notablemente de la segmentación de imágenes [4], ya que, no busca características específicas de un determinado objeto. Esto requiere una interpretación de alto nivel en el contenido de la imagen mientras que la extracción de características es la etapa inicial que tiene como objetivo lograr dicha interpretación. Además, permiten encontrar correspondencias a pesar de grandes cambios en las condiciones de visualización, oclusiones y desorden de imágenes.

4.2.2 Propiedades de las características locales

A partir de dos imágenes de la misma escena captadas en distinta perspectiva o en diferentes condiciones, se considerarán características locales a las regiones que cumplen las siguientes propiedades [3], [5]:

- **Repetibilidad:** un alto porcentaje de las características detectadas deben ser encontradas en las zonas de información común entre las imágenes analizadas.

- **Diferenciación:** los patrones establecidos para las características detectadas deben tener una amplia variación entre sí, lo cual, permite que sean distinguibles y emparejadas correctamente.
- **Localidad:** las características al ser locales reducen la probabilidad de oclusión, lo cual permite establecer aproximaciones de modelos simples de las deformaciones geométricas y fotométricas entre imágenes.
- **Cantidad:** el número de características detectadas deben ser lo suficientemente grande, de tal manera que los objetos pequeños contengan un amplio número de características. Este número se relaciona con el contenido de la información de la imagen, lo cual, provee una representación compacta. Sin embargo, esto depende de la aplicación.
- **Exactitud:** las características detectadas deberán ser localizadas con precisión considerando posibles cambios de localización, escala o forma.
- **Eficiencia:** la detección de características deberá permitir ejecutar aplicaciones conocidas como *de tiempo-critico,* las cuales incluyen aplicaciones en tiempo real.

De estas propiedades, la *repetibilidad* tiene una notable importancia, debido a que, de ella depende que las características cumplan con las propiedades de invariancia y robustez.

- **Invariancia:** cuando las deformaciones que se esperan son grandes, de ser posible se realiza un modelado matemático que permita que la detección de características no se vea afectado por esas transformaciones.
- **Robustez:** en el caso de deformaciones pequeñas, la detección de características puede ser menos sensitiva, en este caso la exactitud disminuye, pero no de manera drástica. Las pequeñas deformaciones incluyen ruido, efectos de la discretización, artefactos entre otros. Cuando se utilizan los modelos matemáticos además de la invariancia se logra mayor robustez.

4.3 Características locales vs. globales

Las características locales invariantes no solo permiten encontrar coincidencias en imágenes que presentan deformaciones geométricas, oclusiones, cambios de perspectiva, sino que también ofrecen una interesante descripción del contenido de la imagen para la recuperación de imágenes [6], las tareas de reconocimiento de objetos o escenas [7], [8], incluyendo a objetos específicos para categorías [1] y para registración de imágenes.

Las características globales, por el contrario, han sido propuestas para describir directamente el contenido de una imagen mediante histogramas de color o variaciones de ellos para reconocimiento de objetos y escenas [9]. Estas características funcionan bien cuando las imágenes presentan colores distintivos y el objetivo es la composición general de la imagen en su conjunto, en lugar de un objeto en primer plano. Se han hecho varios estudios y análisis para medir el rendimiento de estos dos tipos de características tanto para el reconocimiento de objetos como para la recuperación de imágenes [10], [11].

Otros enfoques han sido desarrollados para mejorar el rendimiento de las características globales, entre ellos los *segmentos de imagen* y las *características muestreadas.* Referente a los segmentos de imagen el mejor representante, propuesto por Carson et al. en [12], es conocido como Sistema blobworld, donde los segmentos están basados en color y en textura, mientras que las características muestreadas permiten muestrear de forma exhaustiva diferentes sub-partes de la imagen. Este enfoque está basado en una *ventana deslizante* [13], [14].

4.4 Extractor de características locales

Un extractor de características locales invariantes, en forma general, se basa en las siguientes etapas:

- Encontrar un conjunto de **puntos distintivos** en la imagen.
- Seleccionar automáticamente **la escala** para cada punto distintivo.
- Calcular **un descriptor** para cada una de las regiones seleccionadas.
- **Emparejar** los descriptores locales entre dos o más imágenes.

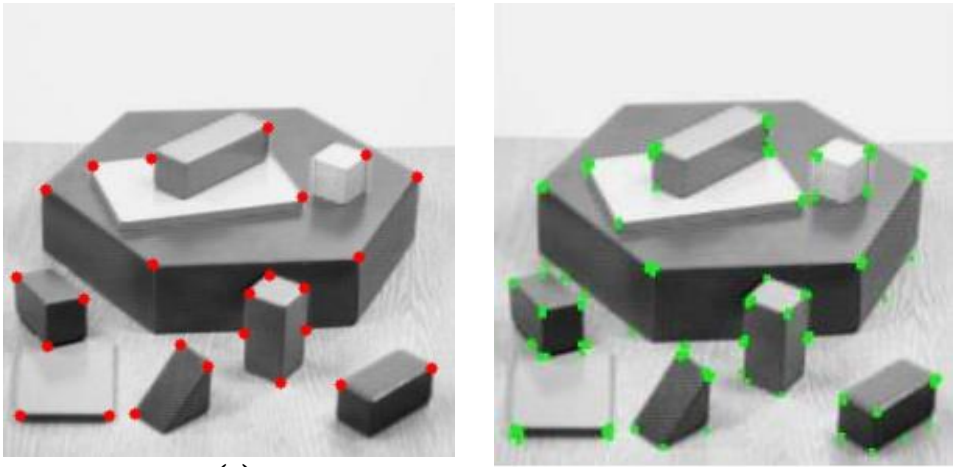
4.4.1 Localización de puntos distintivos

Varios enfoques han sido desarrollados en años recientes con la finalidad de identificar puntos, regiones o bordes en las imágenes, que sean robustos al cambio de escala, rotación, traslación e inclusive frente al ruido y a ciertas distorsiones. Entre estos enfoques se encuentran: la detección de esquinas, la detección de blobs y la detección de regiones [3].

4.4.1.1 Detector de esquinas

En una imagen 2D se considera una esquina a un punto o píxel con amplia curvatura, es decir, un pico de variación local dentro de la imagen, aunque esto no necesariamente se cumple cuando se trata de imágenes 3D. En las esquinas se puede notar que existe la unión de superficies texturizadas y límites de oclusiones. Desde este punto de vista, las esquinas se puede considerar características repetibles en una imagen, las cuales, permanecen invariantes a los cambios mencionados anteriormente.

Varios detectores de esquinas han sido propuestos, entre los cuales los que más resaltan son el detector de Moravec [15], el detector de Harris [16], el detector de Förstner [17] y el detector SUSAN [18]. En la **Fig. 4.2**, se muestra la detección de esquinas mediante dos de estos métodos.



(a) **(b)**

Fig. 4.2: Detección de esquinas. **(a)** Detector de Harris. **(b)** Detector SUSAN.

Cada uno de estos detectores presenta propiedades que los hacen útiles en varias aplicaciones, ya que, por el momento no existe un método general para tareas de visión por computadora. En el caso del detector de Harris permite obtener la invariancia frente a la rotación, pero es muy sensible al ruido. Sin embargo, las características detectadas tienen una alta repetibilidad.

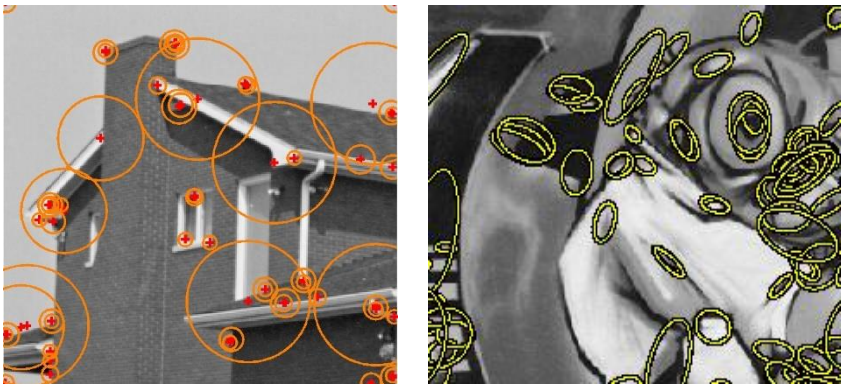
En el caso de SUSAN, las características detectadas no siempre se corresponden con esquinas, sino que muchas veces están relacionadas con los bordes de los objetos contenidos en la imagen. Pero en general, las características detectadas tienen una alta repetibilidad y son inmunes al ruido.

El detector de Harris ha sido utilizado ampliamente en el desarrollo de otras aplicaciones, por ejemplo, Z. Zhang et al. en [19] las esquinas detectadas se utilizan para el alineamiento epipolar de imágenes tomadas en distinta perspectiva [20]. Schmid et al. en [21] utiliza las esquinas para crear un descriptor de imagen local a partir de vectores de orientación-invariante [22].

Sin embargo, ninguno de los detectores anteriormente mencionados, lograron una invariancia frente al cambio de escala de imágenes. Para esto K. Mikolajczyk et al. en [23] propuso el método *Harris-Laplace* para la detección de esquinas o puntos de interés invariantes a la escala.

Este método está basado en la representación del *espacio de escala* presentado por Lindeberg en [24], [25] basado en la propuesta de Young [26], donde se demostró que el filtro Gaussiano y sus derivadas modelan las respuestas del funcionamiento de los campos receptivos en la retina y en la corteza visual de los mamíferos.

Este detector también ha sido desarrollado para extracción de estos puntos de interés cuando las deformaciones son afines. En la **Fig. 4.3**, se muestra la detección mediante este método.



(a) **(b)**

Fig. 4.3: Detección de esquinas y puntos de interés mediante Harris-Laplace. Las regiones encerradas por las circunferencias y elipses corresponden al tamaño de la escala para cada característica. **(a)** Detección para invariancia frente a escala y rotaciones para transformaciones de similaridad. **(b)** Detección para invariancia frente a escala y rotaciones para transformaciones afines.

En la **Fig. 4.4**, se muestra la extracción de características invariantes a la escala en 3 imágenes distintas, las cuales contienen información en común. Un importante aporte a este método ha sido propuesto por Zhang et al. en [27] para mejorar la repetibilidad de las características. Esto tiene importantes aplicaciones en detección y seguimiento de objetos, reconstrucción y registración de imágenes.

(a) **(b)** **(c)**

Fig. 4.4: Detección de características mediante Harris-Laplace **(a)** Imagen de Lena original en escala de grises. **(b)** Imagen de Lena aplicada una rotación positiva y un cambio de escala isotrópico. **(c)** Imagen de Lena aplicada una rotación negativa y un cambio de escala isotrópico.

4.4.1.2 Detector de blobs

Blobs proviene de las iniciales de *Binary Large objects*. Este término está vinculado directamente cuando se trata de tareas de detección de objetos. Un blob es un grupo de píxeles conectados (regiones) de una imagen que tienen un tamaño igual o superior a un umbral establecido previo a la detección. Las regiones que no cumplen con esta condición se descartan, puesto que, se consideran ruido o regiones de no interés [24].

Para esta detección se ha propuesto el detector Hessiano que está basado en la matriz resultante de 2×2 a partir de la expansión de Taylor para la función de intensidad de una imagen $I(x)$ [28]. Este método tiene algunas variantes como el Hessiano-afín y el Hessiano-Laplaciano. La matriz Hessiano se muestra en la **Ec. 4.1**. Otros métodos que han sido desarrollados para esta detección y que tienen una eficiente implementación son: SIFT desarrollado por D. Lowe [29], SURF desarrollado por H. Bay [30] y FAST[8] desarrollado por E. Rosten [31], [32].

$$H = \begin{bmatrix} I_{xx}(x, \sigma_D) & I_{xy}(x, \sigma_D) \\ I_{xy}(x, \sigma_D) & I_{yy}(x, \sigma_D) \end{bmatrix}, \tag{4.1}$$

donde I_{xx}, I_{xy}, I_{yy} corresponden a la segunda derivada de un filtro Gaussiano aplicado en una imagen, σ_D es el valor del espacio de escala definido en [23], [24]. Este filtrado se basa en el determinante y la traza de esta matriz, por lo que es conocido como el Laplaciano (**Fig. 4.5**).

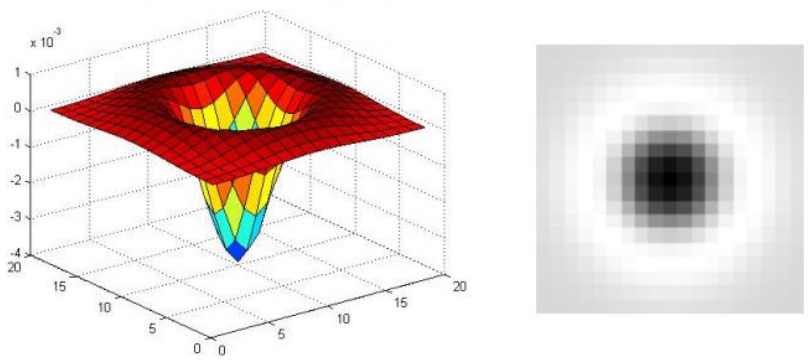


Fig. 4.5: Operador Laplaciano del Gaussiano normalizado utilizado para la detección blob en 2D.

[8] FAST por las siglas en inglés de *Features from accelerated segment test*.

La derivada del filtro Gaussiano permite codificar la información de las estructuras locales de una imagen, mediante el cambio de la normalidad a una iso-superficie lo que permite capturar las propiedades importantes. Los máximos locales resultantes del filtrado se utilizan para detectar las estructuras blob [3].

SIFT optimiza el tiempo de procesamiento con la aproximación del operador Laplaciano mediante la diferencia de filtros Gaussianos (DoG), en un espacio de escalas denominados Octavas. Por el contrario, SURF utiliza la matriz Hessiana para la detección a partir de la convolución con imágenes integrales en un espacio de escalas piramidal [29], [30].

En el caso de FAST, se utiliza un procedimiento distinto para la detección de esquinas, el cual, se basa un círculo de 16 píxeles que permite clasificar si un punto candidato corresponde o no a una esquina [31].

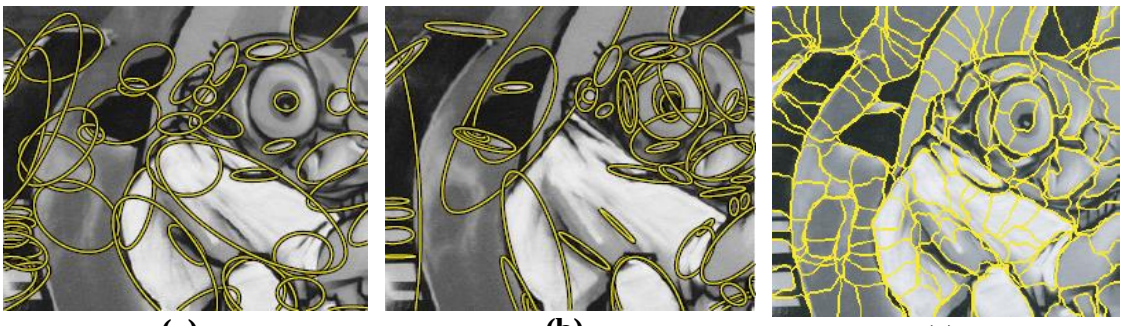
En la **Fig. 4.6**, se muestra la detección blob aplicada en varias imágenes por cada uno de los métodos presentados en esta sección.

(a) **(b)** **(c)**

Fig. 4.6: Detección de estructuras blob en imágenes naturales. **(a)** Detección mediante la matriz Hessian. **(b)** Detección mediante SIFT. **(c)** Detección mediante SURF.

4.4.1.3 Detector de regiones

Una región es una estructura local dentro de la imagen que comparte propiedades similares. Se han propuesto varios métodos para detección de características entre los cuales están: regiones con base en la intensidad [33], regiones extremamente estables MSER[9] [34] y regiones por medio de super píxeles [35]. En la **Fig. 4.7**, se muestra la detección de características mediante los métodos de detección de regiones.



(a) **(b)** **(c)**

Fig. 4.7: Detección de características basado en regiones locales de la imagen. **(a)** Regiones basada en la intensidad. **(b)** Regiones extremamente estables MSER. **(c)** Regiones por super píxeles. **Fuente:** Tuytelaars T. et al., Local Invariant Feature Detectors: A Survey, pp. 242-243, 2008.

[9] MSER por las siglas en inglés de *Maximally Stable Extremal Regions*.

4.4.2 Selección automática de la escala

En visión por computadora, uno de los mayores problemas es que la representación de los objetos del mundo real puede aparecer de distintas formas dependiendo de la escala de observación. Para la interpretación de imágenes desconocidas es crucial contar con una herramienta que brinde una noción de la escala de los objetos de manera automática.

Esto permite hacer frente a las variaciones de tamaño que genera el mapeo en perspectiva, al ruido que se introduce en la formación de imágenes y a la información disponible formada por conjuntos de datos bidimensionales que reflejan las propiedades indirectas del mundo tridimensional [24].

Para la representación a múltiples escalas se utiliza *el espacio de escala, las pirámides y los métodos de distribución no lineal*, las cuales se usan en tareas de preprocesamiento visual como la detección de características, emparejamiento estéreo, flujo óptico, entre otras [36].

4.4.2.1 Espacio de escala

Durante el procesamiento de imágenes no hay forma de conocer *a priori* los valores de escala que permitirán extraer la información relevante, por tanto, una *representación multi escala* de los datos se hace necesaria para encontrar una región característica cuyo tamaño sea covariante con la transformación de la imagen.

Lindeberg postula en [24], [25] que el kernel gaussiano y sus derivadas son los únicos con propiedades para un análisis de espacio de escala debido a su linealidad e invariancia.

Sea una señal $f: \boldsymbol{\mathcal{R}}^D \rightarrow \boldsymbol{\mathcal{R}}$ y la representación del espacio de escala $L: \boldsymbol{\mathcal{R}}^n \times \boldsymbol{\mathcal{R}}_+ \rightarrow \boldsymbol{\mathcal{R}}$**,** donde la representación de f está definida como la solución a la ecuación de difusión. Esto se presenta en la **Ec. 4.2**.

$$\partial_t L = \frac{1}{2}\nabla^2 L = \frac{1}{2}\sum_{i=1}^{D} \partial_{x_i x_i} L \tag{4.2}$$

Con la condición inicial de $L(.;0) = f(.)$. De forma equivalente, esto puede ser definido por la convolución de con varios kernels Gaussianos de amplitud t presentada en la **Ec. 4.3**.

$$L(.;t) = g(.;t) * f(.) \tag{4.3}$$

donde $g: \boldsymbol{\mathcal{R}}^n \times \boldsymbol{\mathcal{R}}_+ \rightarrow \boldsymbol{\mathcal{R}}$ está dado por $g(x,t) = \frac{1}{(2\pi t)^{N/2}} e^{-(x_1^2+\cdots+x_D^2)/2t}$ y $x = (x_1, \ldots, x_D)^T$.

Una propiedad muy conocida de la representación del *espacio de escala* es que la amplitud de las derivadas espaciales, en general, decrece con la escala. Esto significa que cuando una señal está sujeta al suavizado del espacio de escala, se espera que, los datos calculados a partir de las derivadas espaciales disminuyan (**Fig. 4.8**).

Esto es una consecuencia directa de la propiedad de no mejora de los extremos locales, donde el valor en un máximo local no puede aumentar y el valor en un mínimo local no puede disminuir [24].

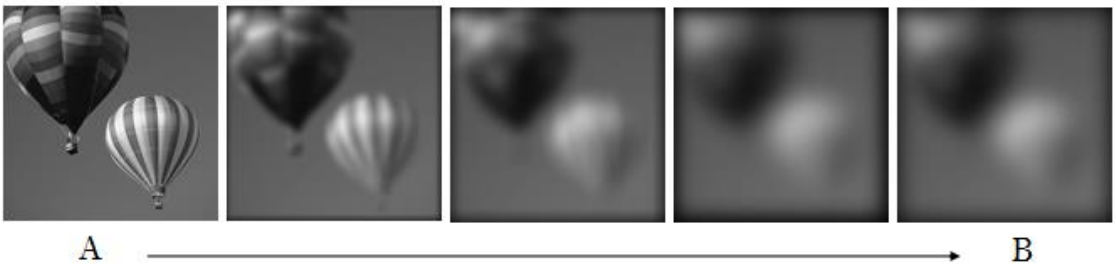


Fig. 4.8: Espacio de escala definido a partir del suavizado de imágenes mediante filtros Gaussianos. **A** corresponde a la imagen de alta resolución hasta llegar a **B** con menos valores de alta frecuencia.

Por tanto, se puede representar una característica en diferentes resoluciones aplicando la función apropiada (combinaciones de derivadas) a diferentes escalas. Para mantener la propiedad de invariancia frente a la escala es necesario normalizar la función derivativa con respecto a la escala de observación. La escala derivativa normalizada de orden *m* se muestra en la **Ec. 4.4** [23].

$$D_{i_1,\ldots,i_m}(x,t) = t^m L_{i_1,\ldots,i_m}(x,t) \tag{4.4}$$

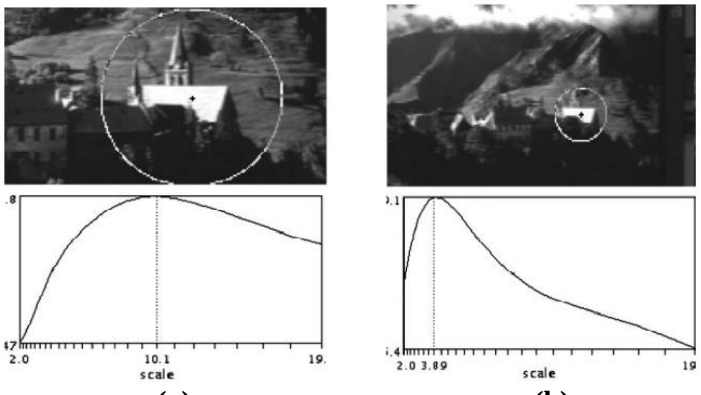
En la **Fig. 4.9**, se presentan dos imágenes de la misma escena captadas en distinta perspectiva con cambio en la escala, donde la relación entre ellas está dada por una transformación geométrica T. Esto se representa de la siguiente manera: $\mathrm{I(x)} = \mathrm{I'(x')}$, donde $\mathrm{x'} = T\mathrm{x}$. Las derivadas normalizadas para este caso están relacionadas de acuerdo con la **Ec. 4.5.**

$$t^m L_{i_1,\ldots,i_m}(x,t) = T^m t^m L_{i_1,\ldots,i_m}(x',Tt) \tag{4.5}$$

Para mantener la información uniforme entre los distintos niveles de resolución el factor de escala se distribuye de manera exponencial. Por tanto, un *espacio de escala normalizado* está determinado por $\mathrm{L(x, t}_n)$ para el valor t_n mostrado en la **Ec. 4.6.**

$$t_n = k^n t_0 \tag{4.6}$$

donde t_0 es el factor de escala inicial en el mejor nivel de resolución, t_n son los niveles sucesivos de la representación del espacio de escala y k es el factor de cambio de escala entre niveles [23].



(a) **(b)**

Fig. 4.9: Selección automática del valor de escala para dos imágenes tomadas en distinta perspectiva. **(a)** La respuesta al Laplaciano Normalizado en el espacio de escala definido es 10.1. **(b)** La respuesta al Laplaciano Normalizado en el espacio de escala definido es 3.89. **Fuente:** Tuytelaars T. et al., Local Invariant Feature Detectors: A Survey, Cap. 3, pp. 223, 2008.

El filtro más utilizado para la selección automática de escala es el *Laplaciano de Gaussiano (LoG)* propuesto por Lindeberg en [25], para la búsqueda del espacio de escala extrema de una escala-normalizada. La **Ec. 4.7** muestra la definición del filtro LoG.

$$L(x,t) = t^2\left(I_{xx}(x,t) + I_{yy}(x,t)\right) \tag{4.7}$$

Lowe en [29] propone la aproximación de LoG mediante la *diferencia de Gaussianas (DoG)* lo cual permite optimizar el tiempo de procesamiento especialmente para aplicaciones de tiempo real (**Fig. 4.10**). La **Ec. 4.8** muestra la definición del filtro DoG.

$$D(x,t) = \big(g(x,kt) - g(x,t)\big) * I(x), \tag{4.8}$$

Las máscaras de estos filtros 2D toman la forma de una región central circular con pesos positivos, rodeada por otra región circular con pesos negativos. Por tanto, la respuesta es más fuerte para las estructuras de imágenes circulares cuyo radio corresponde con la escala del filtro.

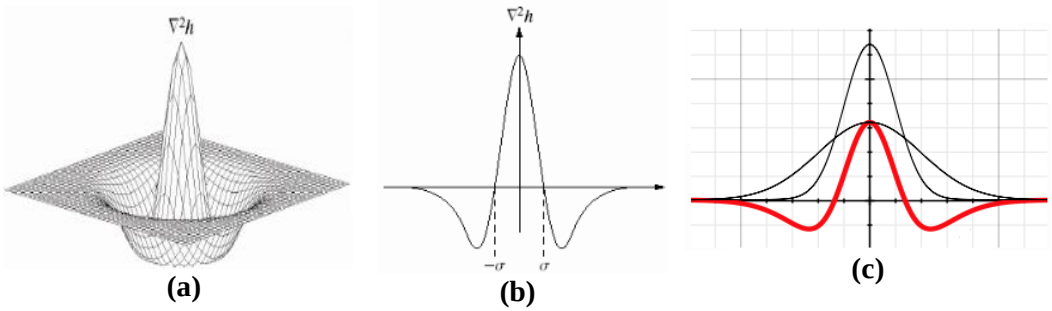


Fig. 4.10: Filtro para selección automática de escala. **(a)** Laplaciano de Gaussiano en 2D. **(b)** Laplaciano de Gaussiano unidimensional. **(c)** DoG obtenido a partir de 2 gaussianas con distinto valor de sigma.

4.4.2.2 Pirámide de imágenes

Una de las estructuras de datos más comunes para representar una imagen de entrada *I* en diferentes tamaños son las pirámides. La capa base de la pirámide está formada por la imagen de entrada y los subsecuentes niveles por las imágenes reducidas de tamaño [37], como se muestra en la **Fig. 4.11.**

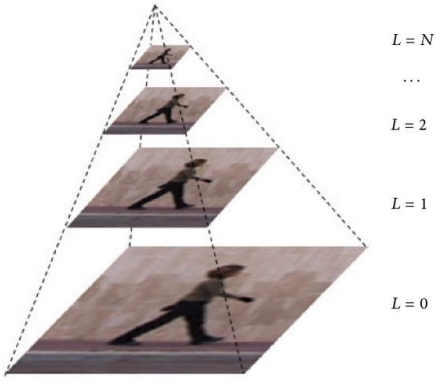


Fig. 4.11: Representación visual de una pirámide de imágenes con 4 niveles (L=4).

Las pirámides se han utilizado como el tipo principal de representación a múltiples escalas para la extracción de características a partir de imágenes del mundo real, ya que expresan aproximaciones computacionalmente eficientes para la representación del espacio de escala [38].

Las pirámides están clasificadas en Gaussianas, Laplacianas y dirigidas. Estas son utilizadas en diferentes tareas de visión por computadora, entre ellas, la compresión de imágenes, análisis de textura y reconocimiento de objetos [29], [30], [39], [40].

4.4.3 Descriptores de imágenes

Un descriptor es un vector de características que se crea a partir de componentes del mismo tipo, el cual mide las propiedades estadísticas, geométricas, algebraicas, espaciales o diferenciales de una imagen. Por tanto, un descriptor es una representación compacta de información que permite distinguir una imagen de otra [41].

Siguiendo con el desarrollo de un extractor de características, una vez que se ha identificado el conjunto de puntos distintivos junto con las regiones de interés, éstas se codifican mediante un descriptor adecuado para una comparación discriminativa entre ellas. Este es un paso importante en la registración de imágenes, puesto que, son los descriptores los que permiten determinar la correspondencia entre las características encontradas en dos imágenes, para posteriormente, calcular la matriz de transformación que reduzca la diferencia entre las imágenes analizadas [41].

Entre los descriptores más utilizados en la extracción de características se pueden mencionar los siguientes: basados en histogramas, descriptor SIFT y sus variantes, basados en imágenes de espín, basados en filtrado, basados en momentos y los descriptores compuestos. Varios estudios se han realizado en torno al análisis del rendimiento de estos descriptores [42], [43].

4.4.3.1 Descriptor basado en histogramas

El histograma es usado para caracterizar la distribución de color de una imagen. Cuando se calcula el histograma de intensidad en una ventana circular se obtiene un vector rotacionalmente invariante que describe la información contenida en dicha ventana.

El histograma es robusto frente a pequeños cambios en la perspectiva y en la escala, incluso cuando existen oclusiones en los objetos de interés. Sin embargo, la desventaja que presenta es que, dos imágenes distintas pueden producir histogramas similares [41].

De manera general, una imagen está representada en el espacio de color RGB[10], por tanto, el histograma contendrá información de las 3 componentes de dicho espacio. Para el descriptor basado en el histograma de color se realiza una cuantización, especialmente cuando las imágenes analizadas son muy pequeñas. Esta cuantización puede causar que dos colores similares sean agrupados en distintos bins del histograma. Para evitar esto, una Gaussiana es centrada en el bin del histograma correspondiente al color del píxel y se analizan de manera inversamente proporcional a su distancia al bin [41], [44]. Esto se muestra en la **Ec. 4.9**.

$$H(R,G,B) = \sum_{x}\sum_{y} G_{\sigma}(R(x,y), G(x,y), B(x,y)) \tag{4.9}$$

donde G_{σ} es una gaussiana 3-D con desviación estándar σ centrada en el bin (R,G,B) y $R(x,y), G(x,y), B(x,y)$ son las componentes en los 3 canales del píxel (x,y). La suma está dada sobre todos los píxeles incluidos en la ventana circular de radio r centrada en el bin del histograma.

[10] Espacio de color representado por los colores primarios R (*red*), G (*green*) y B (*blue*). Cada píxel de una imagen color tiene valores entre [0, 255] por cada canal.

En la **Fig. 4.12**, se muestra un ejemplo sobre el descriptor basado en histogramas de color y su diferencia con el descriptor SIFT que se detalla en la siguiente sección. Para el descriptor de histogramas de color se puede convertir la imagen del espacio de color RGB a VSH[11], donde se utiliza el valor y la saturación para mejorar la distribución del rango de los colores.

Existen diversas propuestas para los descriptores basados en el histograma, entre ellos, descriptor basado en los histogramas de bordes [45], [46] y descriptores por histograma de gradientes HoG [47], [48]. Se utilizan en aplicaciones de reconstrucción de imágenes o en detección de objetos.

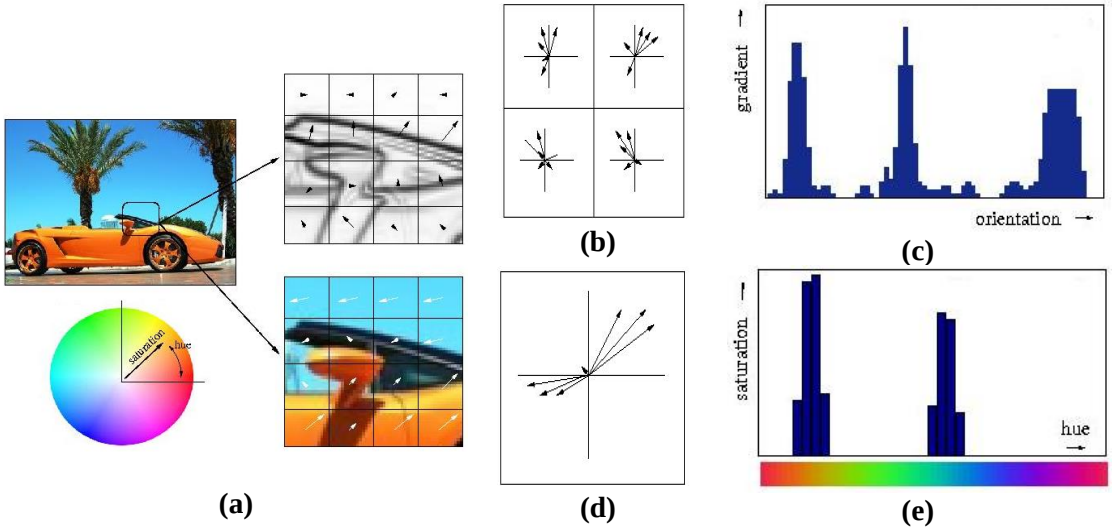


Fig. 4.12: Descriptor basado en histogramas de color y SIFT. **(a)** Región de interés a ser descrita. Para el descriptor de color se utiliza el espacio HSV. **(b)** Generación de 4 bins para la orientación del descriptor SIFT de acuerdo con la magnitud y orientación del gradiente. **(c)** Distribución del histograma SIFT. **(d)** Generación de las direcciones de saturación de color. **(e)** Distribución del histograma de color. **Fuente:** http://lear.inrialpes.fr/people/vandeweijer/color_descriptors.html.

4.4.3.2 Descriptor SIFT

A cada característica detectada, por medio de SIFT, se le asigna una orientación local específica, de acuerdo con las propiedades del entorno dentro de la imagen para lograr la invariancia frente a la rotación. Para esto, las imágenes obtenidas en el espacio de escala L se utilizan para calcular la magnitud y orientación del gradiente, como se muestra en la **Ec. 4.10**.

$$m(x,y) = \sqrt{\big(L(x+1,y) - L(x-1,y)\big)^2 + \big(L(x,y+1) - L(x,y-1)\big)^2}$$

$$\theta(x,y) = tan^{-1}\left(\frac{\big(L(x,y+1) - L(x,y-1)\big)}{\big(L(x+1,y) - L(x-1,y)\big)}\right) \qquad (4.10)$$

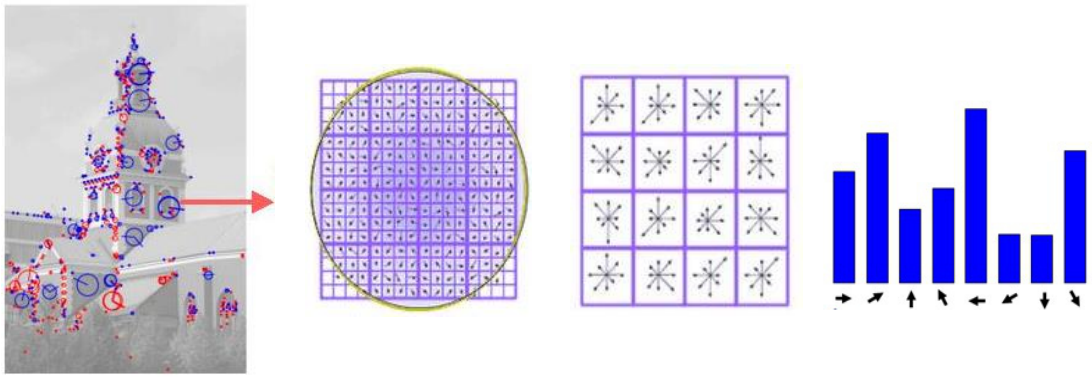
donde $L(x,y)$ es el espacio de escala para la característica en la posición (x,y), $m(x,y)$ es la magnitud del gradiente y $\theta(x,y)$ es el valor de la orientación.

El histograma se forma a partir de las orientaciones del gradiente alrededor de la *característica* y la región establecida por el espacio de escala. El histograma contiene 36 bins para cubrir los 360 grados. Cada muestra que se agrega al histograma es ponderada por su magnitud de gradiente y

[11] Espacio de color representado además del matiz por la saturación y el valor para obtener el valor del color. Sus siglas provienen de H (*Hue*), S (*Saturation*) y V (*Value*).

por una ventana circular gaussiana con una sigma igual a 1.5 veces el valor de la escala de la característica [29].

Los picos en el histograma corresponden a las direcciones dominantes de los gradientes locales. Entonces se detecta el pico más alto en el histograma, y luego se usa otro pico local que esté dentro del 80% del pico más alto para crear una característica con esa orientación [22], [29]. Este proceso se resume en la **Fig. 4.13.**

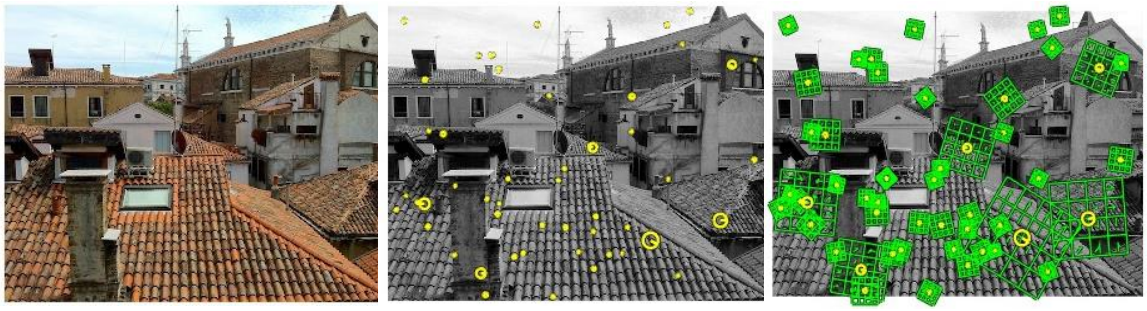


(a) **(b)** **(c)** **(d)**

Fig. 4.13: Descriptor local SIFT. **(a)** Extracción de características con su región de acuerdo con el valor de escala automática. **(b)** Calculo de la magnitud y orientación del gradiente ponderados de acuerdo con una ventana circular Gaussiana. **(c)** Histograma acumulado de dimensión 4×4. **(d)** Detección de la orientación predominante para la característica. **Fuente:** https://medium.com/machine-learning-world/feature-extraction-and-similar-image-search-with-opencv-for-newbies-3c59796bf774.

El descriptor se forma a partir de un vector que contiene los valores de todas las entradas del histograma de orientación, correspondientes a las longitudes del gradiente. En la **Fig. 4.12(c)** se muestra una serie de histogramas de orientación de 4×4, con 8 bins de orientación en cada uno. Por tanto, el vector de características está formado por $4 \times 4 \times 8 = 128$ elementos para cada característica [29].

Finalmente, el vector de características se normaliza a la unidad de longitud para reducir los efectos del cambio de iluminación tanto en referencia al contraste como al brillo de la imagen. En la **Fig. 4.14**, se muestra la aplicación de SIFT en la extracción de características.



(a) **(b)** **(c)**

Fig. 4.14: Extracción de características mediante SIFT. **(a)** Imagen natural original en espacio color RGB. **(b)** Identificación de puntos distintivos y cálculo de sus valores de escala en la imagen en escala de grises. **(c)** Aplicación del descriptor basado en el histograma de gradiente. **Fuente:** https://www.ics.uci.edu/~majumder/VC/211HW3/vlfeat/doc/overview/sift.html.

4.4.3.3 Descriptor basado en momentos

Los momentos invariantes se han utilizado ampliamente en imágenes para el reconocimiento de patrones, debido a sus características de invariancia frente a las deformaciones lineales. Su teoría tiene base en la física y se conocen también como momentos geométricos invariantes, ya que permiten describir la geometría de un patrón dentro de una ventana circular [49].

Introducidos inicialmente por Hu [50] quien derivó seis invariantes ortogonales absolutos y un invariante ortogonal sesgado basado en invariantes algebraicos, los cuales, son independientes de la posición, escala y orientación junto con la proyección paralela. Actualmente los momentos invariantes más conocidos son: momentos geométricos [51], [52] momentos de Zernike [53], momentos rotacionales [54] y momentos complejos [55].

En el caso de los momentos de Zernike, utilizados en la propuesta de un método para extracción de características, se asume a la imagen en coordenadas polares (ρ, θ), donde el valor de la intensidad del píxel está dado por $f(\rho, \theta)$, medidas desde el centro de una ventana circular de radio en el rango $[0, 1]$ utilizada para normalización (**Fig. 4.15**).

Se asume A_{nm} como el momento de Zernike, donde (n, m) es el orden del polinomio. El mismo momento rotado en el centro de la ventana un ángulo α será $A_{nm}\exp(jn\alpha)$, donde $j = \sqrt{-1}$. Si dos valores de momentos de Zernike describen dos ventanas rotadas un ángulo α, los vectores de descripción no tendrán la diferencia de fase, aunque representen distintos patrones. Esto permite a los momentos de Zernike establecer la correspondencia entre ventanas circulares en dos imágenes y además hace posible determinar la diferencia de rotación entre ellas [41].

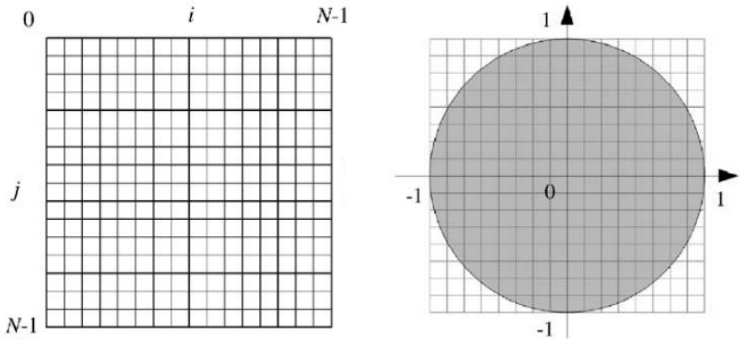


Fig. 4.15: Normalización para cálculo de momentos de Zernike. **(a)** Imagen o región de tamaño $N \times N$. **(b)** Normalización de la información con una ventana circular de radio $r = 1$.

La forma discreta de los momentos de Zernike para una imagen o región dentro de ella de tamaño $N \times N$, se calculan como lo muestra la **Ec. 4.11**.

$$Z_n^m = \frac{n+1}{\lambda_N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) R_n^m(\rho_{xy}) exp\,(-jm\theta_{xy}) \qquad (4.11)$$

donde n es el orden del polinomio de Zernike, m es el número de repeticiones, N el tamaño de la imagen a describir, λ_N es un factor de normalización y es igual al número de píxeles localizados dentro de la ventana circular de radio 1 que han sido mapeados por el descriptor, $f(x, y)$ es el valor de intensidad de los píxeles, $\mathrm{R_n^m}$ son los polinomios radiales de Zernike, $0 \leq \rho_{xy} \leq 1$ es la distancia transformada y θ_{xy} es la fase [56].

Los momentos de Zernike presentan las siguientes ventajas [57]:

- Las magnitudes de los polinomios son invariantes a la rotación.
- Son robustos al ruido y a pequeños cambios en la geometría de los objetos descritos.
- Dado que tienen base ortogonal, la redundancia de información es mínima.

Sin embargo, estos momentos también presentan algunas desventajas, entre ellas está que se requiere una normalización del espacio de coordenadas para que se defina el dominio ortogonal, lo cual, requiere el cambio de las integrales continuas a sumatorias discretas. Esto podría conducir a pequeños errores de aproximación y a complejidad computacional. Sin embargo, se han hecho diversas propuestas para realizar un cálculo más rápido manteniendo la precisión [56], [58].

De acuerdo con el orden y al número requerido de los momentos de Zernike, se tienen los descriptores que se muestran en la **Fig. 4.16.**

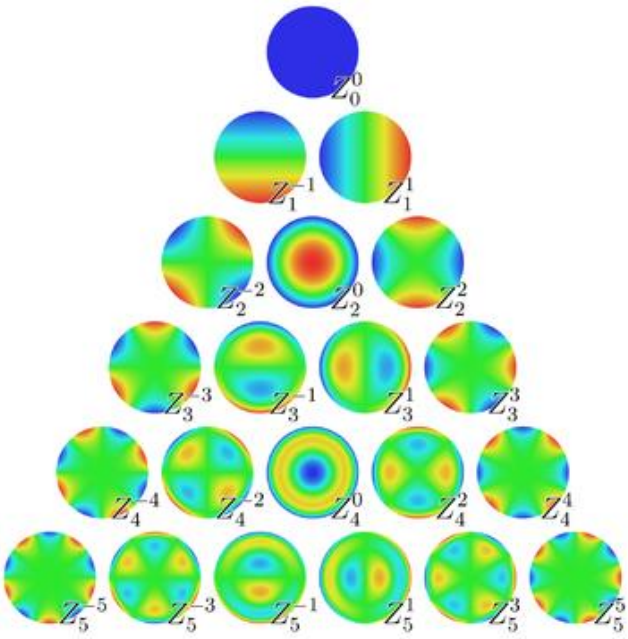


Fig. 4.16: Componentes reales de los momentos de Zernike de orden $n = 5$ y repeticiones $m = 5$.

En general, los momentos invariantes tienen aplicaciones en el reconocimiento de patrones [59], procesamiento de imágenes [60], extracción de características incluyendo transformaciones afines [61], reconstrucción de imágenes [62], [63], registración de imágenes [64] y estimación de movimiento [65].

4.4.4 Emparejamiento de descriptores

Una vez que las características han sido descritas y se ha obtenido un vector vinculado a la información de la región seleccionada, se procede a emparejarlos cuando las tareas están relacionadas con registración de imágenes, reconocimientos y seguimiento de objetos.

Normalmente se utiliza la distancia Euclídea para encontrar el mejor candidato, por tanto, las características que tengan la menor distancia serán emparejadas [29]. Sin embargo, debido a que pueden existir cambios de perspectiva u oclusiones, muchas veces esta métrica puede no ser la más adecuada. Hisham et al. en [66] usa la *Suma de diferencias cuadráticas (SSD)* como métrica para el emparejamiento de características.

Esta es una métrica de distancia (**Ec. 4.12**) que requiere un valor de umbral para aceptar o descartar el emparejamiento.

$$SSD = |f_1 - f_2|^2$$

$$Relacion\ de\ distancia = \frac{SSD(f_1, f_2)}{SSD(f_1, f_2')} \tag{4.12}$$

donde f_1, f_2 y f_2' son las características que menor distancia Euclídea presentan. Por tanto, para encontrar el mejor emparejamiento se calcula la relación de distancia *SSD* entre ellas.

La relación *SSD* está en el rango $[0.0, 1.0]$ y se define un umbral para establecer cuando dos características serán emparejadas. Esto se muestra en la **Fig. 4.17**.

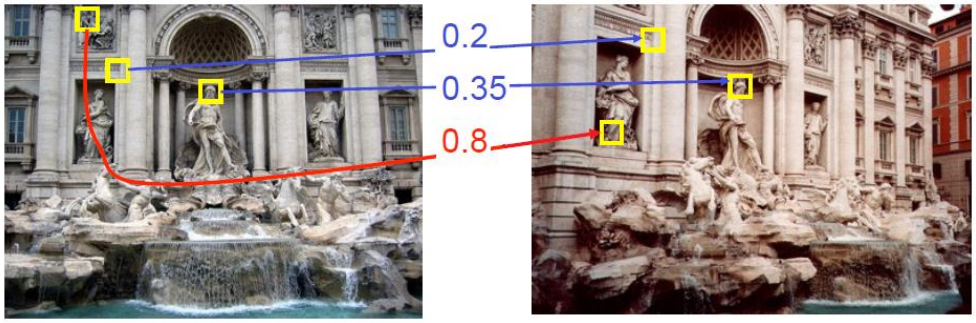


Fig. 4.17: Emparejamiento de características entre dos imágenes. Valores obtenidos con la relación de distancia SSD. **(a)** Valores en color azul están por debajo del umbral establecido por tanto el emparejamiento es correcto. **(b)** Valor en color rojo supera el umbral, el emparejamiento no es aceptado.

Aplicando estos extractores de características en imágenes médicas, se puede obtener el resultado mostrado en la **Fig. 4.18.** Las imágenes son resonancias magnéticas MRI de cerebro en distinta modalidad.

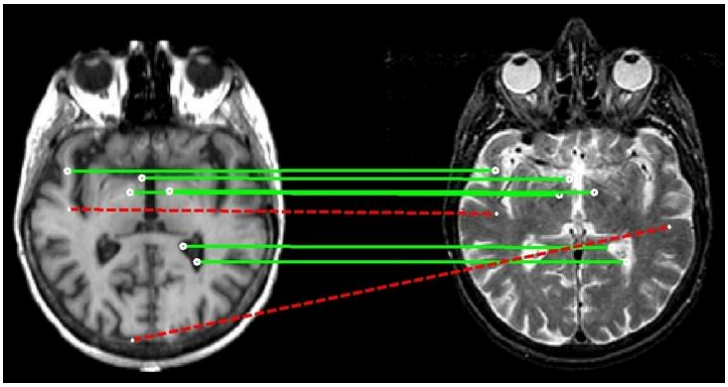


Fig. 4.18: Extracción de características con SIFT en imágenes MRI de cerebro en modalidades T1 y T2.

4.5 Registración basada en características

Como se mencionó en el **Capítulo 2**, la registración de imágenes tiene como objetivo calcular la mejor matriz de transformación T para reducir la diferencia geométrica entre dos imágenes. Cuando ésta es abordada desde la perspectiva de la extracción de características, se tendrán dos conjuntos de datos que denominaremos $x = \{f_1, f_2, \dots, f_n\}$ y $x' = \{f_1', f_2', \dots, f_n'\}$ donde x son las características de la imagen de referencia, x' las características de la imagen móvil y n es el número total de características emparejadas.

Con esta información se calculará la *Homografía 2D* definida del siguiente modo: dado un conjunto de puntos x_i y un correspondiente conjunto de puntos x_i' en el plano proyectivo $\mathbb{P}^2$, tal que $x_i \leftrightarrow x_i'$, entonces se calcula la transformación proyectiva H, por tanto $Hx_i = x_i'$ donde $i = (1, .., n)$.

Este proceso se muestra en la **Fig. 4.19,** la cual, es presentada por Hartley et al. en [20]**.**

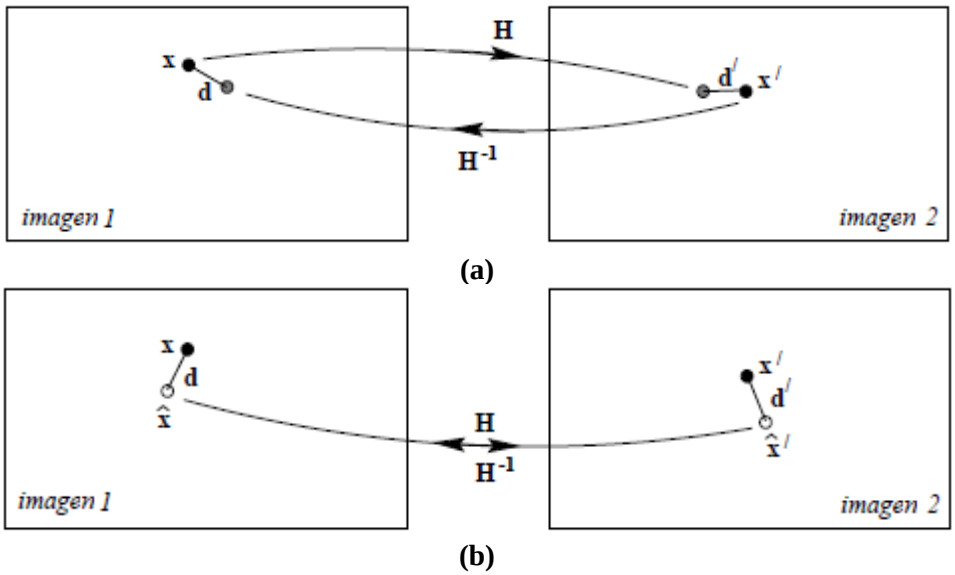


Fig. 4.19: Cálculo de la matriz de transformación H entre las características obtenidas en dos imágenes distintas. **(a)** La transferencia de error simétrica utilizada durante el cálculo de la matriz. **(b)** La utilización de la retroproyección del error para mejorar el cálculo de la transformación. **Fuente:** Harley R. et al., Multiple View Geometry in computer visión, Cap. 4, pp. 92, 2004.

Como se mencionó en la **Sección 2.5** una matriz proyectiva tiene la forma de la **Ec. 4.13**:

$$H = \begin{bmatrix} A & t \\ \mathsf{V}^T & v \end{bmatrix} \tag{4.13}$$

donde A es la matriz de transformación lineal no singular, t es el vector de traslación, $\mathsf{V} = (v_1,\ v_2)^T$ y v el factor de referencia al plano, generalmente $v = 1$.

Esta matriz tiene un tamaño de 3×3 tanto para la transformación proyectiva como para el resto de las transformaciones lineales (**Tabla 2.1**). La transformación proyectiva tiene 8 dof (grados de libertad) que está vinculado con el número de incógnitas a calcular en la matriz H, esto se muestra en la **Ec. 4.14.**

$$H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & 1 \end{bmatrix} \tag{4.14}$$

Las transformaciones de similaridad presentan 3 dof con el vector $\mathsf{V} = (0,\ 0)^T$ (**Ec. 4.13**), donde las incógnitas representan el valor de escala, traslación y rotación.

Dependiendo los grados de libertad de la transformación geométrica que se esté abordando, hay que tener en cuenta que se requiere un número mínimo de puntos para poder realizar este cómputo. Para el caso de la transformación proyectiva se requieren 4 puntos a partir de los conjuntos x_i y x_i'.

4.5.1 Algoritmo de transformación lineal directa (DLT)

El cálculo de la matriz de transformación geométrica H en el dominio $2D \rightarrow 2D$, mediante la correspondencia de puntos (características) $x_i \leftrightarrow x_i'$ está dado por $x_i' = Hx_i$. Se asume que las coordenadas de los puntos característicos están expresadas en la forma homogénea (mencionadas en el Cap. 2).

Para que la ecuación $x_i' = Hx_i$ sea resuelta de manera lineal aplicando el algoritmo DLT [20], [67] es necesario expresarla mediante el producto cruz, como se muestra en la **Ec. 4.15**.

$$x_i' \times Hx_i = 0 \tag{4.15}$$

Considerando que son conocidas las coordenadas geométricas de cada punto en los dos conjuntos x_i y x_i', la forma homogénea esta expresada de la siguiente forma: $x_i' = (\mathrm{x}_i', \mathrm{y}_\mathrm{i}', \mathrm{w}_\mathrm{i}')$ donde $\mathrm{w} = 1$ es el factor de regularización de la escala. Por tanto, en la **Ec. 4.15** la incógnita son los parámetros de la matriz H, la cual puede expresarse como la ecuación lineal: $\boldsymbol{A_i h = 0}$. En la **Ec. 4.16** se muestra el desglose de esta ecuación.

$$\begin{bmatrix} 0^T & -w_i'\mathrm{x}_i^T & y_i'\mathrm{x}_i^T \\ w_i'\mathrm{x}_i^T & 0^T & -x_i'\mathrm{x}_i^T \end{bmatrix} \begin{pmatrix} \boldsymbol{h_1} \\ \boldsymbol{h_2} \\ \boldsymbol{h_3} \end{pmatrix} = 0 \tag{4.16}$$

donde h es la matriz que contiene las filas de la matriz $H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & 1 \end{bmatrix}$ y el tamaño de la matriz A_i es de 2×9.

El proceso de transformación directa lineal DLT se resume en el **Algoritmo 4.1** [20].

Algoritmo 4.1: Transformación directa lineal DLT.

- Seleccionar $n \geq 4$ puntos del conjunto de características emparejadas $x_i \leftrightarrow x_i'$ y proceder a calcular la matriz H tal que $x_i' = Hx_i$.
- Normalizar los datos tanto de x_i como de x_i'. Esta normalización consiste en el cálculo de la transformación de similaridad con un centro en el punto $(0, 0)^T$ y una distancia promedio al origen de $\sqrt{2}$.
- Ensamblar la matriz A de dimensión $2\ n \times 9$. (**Ec. 4.16**)
- Obtener el valor singular más pequeño del vector de unidad singular, mediante el proceso SVD (Descomposición en valores singulares) [68]. Si $A = UDV^T$, h es la última columna de la matriz V con D matriz diagonal con entradas diagonales positivas.
- Aplicar la matriz H calculada a todo el conjunto de características $x_i \leftrightarrow x_i'$.

4.5.2 Métricas de distancia

El proceso de calcular la matriz de transformación H incluye una función de costo. Esta función está basada en la distancia algebraica o en la geométrica que los puntos característicos presentan previa y posterior a la transformación. Como en todo proceso de optimización, se espera calcular la mejor matriz de transformación, la cual, minimice la función de costo propuesta.

4.5.2.1 Distancia algebraica

El algoritmo DLT puede usar como función de costo para minimizar a la distancia algebraica por medio de la norma $\|\boldsymbol{Ah}\|$. Esta distancia es el valor escalar del *vector de error algebraico* $\boldsymbol{\epsilon}_i$ definido como $\epsilon = \mathrm{Ah}$. Cada componente del conjunto $x_i \leftrightarrow x_i'$ contribuye al vector de error $\boldsymbol{\epsilon}_i$, el cual una vez completado permite la estimación de la distancia algebraica [20].

Dado un conjunto de correspondencias, el error $\epsilon = \mathrm{Ah}$ para el conjunto completo está dado por la **Ec. 4.17**, que muestra la definición de la distancia algebraica.

$$\sum_i d_{alg}(x_i', Hx_i)^2 = \sum_i \|\epsilon_i\|^2 = \|Ah\|^2 = \|\epsilon\|^2 \tag{4.17}$$

4.5.2.2 Distancia geométrica

Es la distancia Euclídea medida entre los puntos del conjunto x_i' y los valores transformados a partir de la matriz H denominados $\overline{x_i}$. Esta distancia se puede calcular con base a una de las dos imágenes, lo cual, para casos prácticos puede presentar ciertos inconvenientes. Por tanto, la mejor matriz de transformación se calcula minimizando esta función de costo. En la **Ec. 4.18**, se muestra la definición de la distancia geométrica [20].

$$\sum_i d(x_i', H\overline{x_i})^2 \tag{4.18}$$

4.5.3 Transferencia sistemática y retroproyección del error

En la **Fig. 4.18**, se muestra los dos métodos utilizados para el cálculo de la matriz de transformación [20].

Transferencia sistemática del error

Como se mencionó en el caso de la distancia geométrica, esta puede ser calculada con base en una de las dos imágenes (generalmente la imagen móvil), pero para casos prácticos es mejor calcular la función de costo teniendo en cuenta las dos imágenes simultáneamente.

Para esto se tiene en cuenta tanto la matriz H como su inversa H^{-1}, donde la suma geométrica del error está dada con base en las dos transformaciones como lo muestra la **Ec. 4.19.**

$$\sum_i d(x_i, H^{-1}x_i')^2 + d(x_i', Hx_i)^2 \tag{4.19}$$

Retroproyección del error

Otro método alternativo para el cálculo del error considerando las dos imágenes está basado en la retroproyección de error. En este método se estima una *corrección* para cada correspondencia de los conjuntos $x_i \leftrightarrow x_i'$. Mediante este proceso, se calcula la matriz H con pares $\hat{x}_i, \hat{x}_i'$ perfectamente emparejados (**Fig. 4.18 (b)**). Esto se muestra en la **Ec. 4.20**.

$$\sum_i d(x_i, \hat{x}_i)^2 + d(x_i', \hat{x}_i')^2 \tag{4.20}$$

donde $\hat{x}_i' = \hat{H}\hat{x}_i$ para todo i.

4.5.4 Algoritmo RANSAC

RANSAC, presentado por Foley et al. en [69], es un algoritmo iterativo robusto utilizado para la estimación de los parámetros de un modelo matemático a partir de un conjunto de datos. En este algoritmo, la estimación no es afectada por los datos atípicos (*outliers*) o ruido que puedan contener dichos datos. En la **Fig. 4.20** se muestra el ajuste de un conjunto de datos a un modelo lineal.

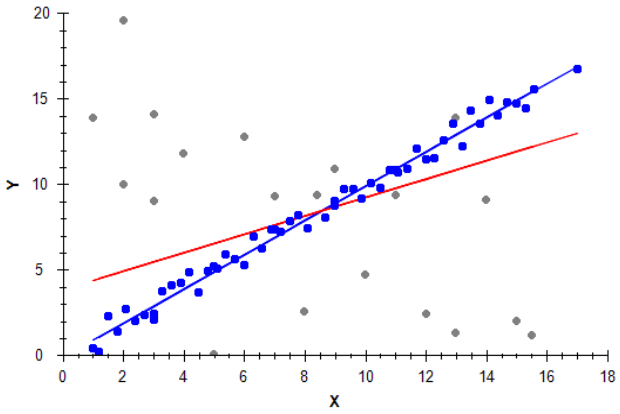


Fig. 4.20: Ajuste de un modelo lineal en 2D. **(Color rojo)** Modelo obtenido con el método SIMPLEX con el conjunto total de datos. **(Color azul)** Modelo obtenido con RANSAC, los puntos en color azul se denominan inliers y los puntos en gris son los outliers descartados para el cálculo de los parámetros deseados. **Fuente:** https://github.com/philipperemy/Ransac-Java.

Este algoritmo es considerado uno de los primeros diseñados específicamente para el área de *Visión por computadora*. Se ha utilizado en el cálculo de la homografía o la transformación proyectiva entre dos imágenes a partir de un conjunto de características emparejadas.

La idea es utilizar la menor cantidad de puntos posibles para estimar el modelo y luego ver cuántos datos se ajustan al modelo estimado. Para el caso de estimar una homografía H se eligen aleatoriamente $n = 4$ puntos dentro del conjunto $x_i \leftrightarrow x_i'$ y se calcula la homografía aproximada $\widehat{H}$. Con $\widehat{H}$ se cuantifica la cantidad puntos son compatibles con la distancia establecida como umbral. Si esta cantidad es suficiente, se utilizan estos puntos para estimar el mejor valor de la matriz de transformación $\widehat{H}$ [20], [69].

RANSAC se resume en el **Algoritmo 4.2.**

Algoritmo 4.2: RANSAC.

- Dado un modelo que requiere un mínimo número de puntos n para determinar sus parámetros, se tiene un conjunto de datos P del cual se eligen aleatoriamente los n puntos para instanciar el modelo.
- Con el modelo instanciado M_1 se determina el subconjunto de decisión S_1 de puntos que están a una distancia menor igual que t del modelo M_1.
- Si la cantidad de puntos en S_1 es mayor que un umbral T entonces se elige el subconjunto de decisión S_1 para computar el nuevo modelo M_1^*.
- Si la cantidad de puntos en S_1 es menor que T, se elige un nuevo subconjuntoS_2 y se repite el proceso.

El proceso se realiza un numero N de iteraciones y se resuelve el modelo con el subconjunto de decisión que contuvo la mayor cantidad de puntos (*inliers*).

4.6 Contribuciones originales

Las contribuciones de este capítulo están enfocadas a la propuesta de un algoritmo para extracción de características, con un enfoque específico en las imágenes médicas y su posterior aplicación en la registración. Como se mencionó en el **Capítulo 2**, las imágenes médicas presentan importantes diferencias frente a las imágenes naturales.

4.6.1 Extractor de características invariantes

Dentro del campo médico, a las características locales relevantes de una imagen se las denomina *marcadores anatómicos*. Estos marcadores pueden ser identificados tanto de manera automática como manual. En este último caso, el procedimiento lo ejecuta un especialista, generalmente, un profesional médico.

Para la forma automática existen diferentes técnicas utilizadas para reducir el error tanto en la identificación del marcador como en la registración final. Una de las técnicas más conocidas es mediante la extracción de *puntos o marcadores fiduciarios* [70], [71]. Los marcadores fiduciarios utilizan un objeto ubicado dentro del campo de visión del sistema de adquisición de imágenes para que aparezca en la imagen producida y sea utilizado como un punto de referencia. Este método ha sido ampliamente utilizado en la registración basada en puntos [72], [73].

El enfoque que será propuesto a continuación se basa en la extracción automática de marcadores anatómicos basados en la información de la imagen que tengan las propiedades mencionadas en la sección 4.2.2. Por tanto, las regiones locales extraídas deben ser, entre otras propiedades, distinguibles, invariantes y repetibles. En Isa-Jara et al. [74] se presenta el aporte realizado de esta sección.

4.6.1.1 Identificación de puntos distintivos

Las imágenes médicas presentan, en general, regiones con alta homogeneidad y bajo contraste, por lo que para su procesamiento es necesario contar con un filtro robusto que permita discriminar adecuadamente la información existente en las diferentes regiones. El filtro de Gabor ha sido utilizado con diversas aplicaciones debido a sus características especialmente por su versatilidad en el cambio de escala y orientación.

Este filtro, propuesto por Dennis Gabor [75], es de tipo lineal cuya respuesta al impulso es una función sinusoidal multiplicada por una Gaussiana, lo que permite detectar regiones con alta respuesta de energía. El filtro de Gabor ha recibido una considerable atención debido a está relacionado con el funcionamiento de las neuronas de la corteza visual primaria [76], por lo que, ha sido utilizado en aplicaciones como reconocimiento facial, segmentación de textura y detección de bordes [77].

El banco de filtros de Gabor es usado para detectar las características conocidas como *características de Gabor* para una imagen de entrada [78]. El banco de filtros permite realizar un análisis multi-resolución mediante un conjunto de filtros a diferentes escalas (frecuencias) y orientación (ángulos) para asegurar la invariancia, por tanto, los objetos analizados pueden ser reconocidos cuando haya un cambio en el escala, rotación y traslación. Además, este banco de filtros permite definir el espacio conocido como *espacio de características de Gabor* [75].

El filtro de Gabor normalizado se calcula de acuerdo con la **Ec. 4.21.**

$$gb(x,y) = \frac{f^2}{\pi\gamma\eta}\,\mathbf{exp}^{-\left[\frac{x'^2f^2}{\gamma^2}+\frac{y'^2f^2}{\eta^2}\right]}\,\boldsymbol{e}^{j2\pi fx'} \tag{4.21}$$

donde $x' = x\cos\theta + y\sin\theta,\ y' = -x\sin\theta + y\cos\theta$, x, y son las coordenadas de las escalas en el sistema de referencia, x', y' en el sistema de referencia θ, f representa la distancia desde el origen al centro de la función gaussiana, θ la orientación de la función de Gabor, γ y η caracterizan la nitidez de la función gaussiana a lo largo de los ejes mayor y menor [79].

En la **Fig. 4.21**, se muestra la parte real de un banco de filtros de Gabor con escalas $m = 5$ y orientaciones $n = 8$, utilizadas en diversas aplicaciones.

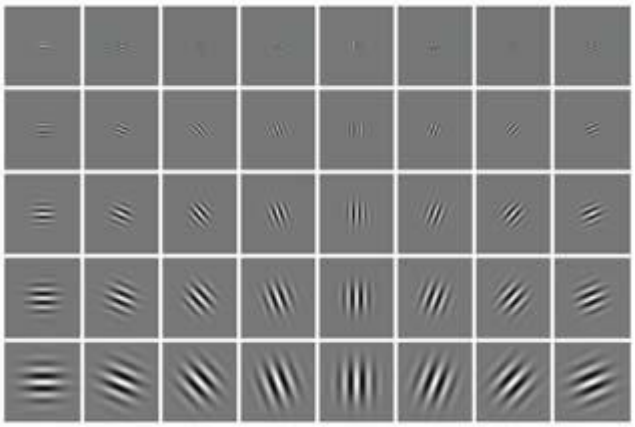


Fig. 4.21: Banco de filtros de Gabor normalizado con $m = 5$ escalas (frecuencias) y $n = 8$ orientaciones.

Además, el filtro de Gabor mediante a su componente Gaussiana permite definir un espacio de escalas para su posterior análisis piramidal. El espacio de escala será definido como: $\sigma_{1\ldots m} = k^{1\ldots m}\sigma_0$, donde σ_m es el valor de la escala o frecuencia a partir del valor inicial σ_0 multiplicado k, el cual es el parámetro de cambio de escala del conjunto de niveles $[1\ldots m]$.

Para la detección de puntos distintivos se utilizan solamente 2 orientaciones $[0^\circ, 90^\circ]$ para evitar la redundancia de la información a la salida del filtro. En la **Fig. 4.22** se muestra las orientaciones del filtro de Gabor utilizadas para la detección.

(a) **(b)**

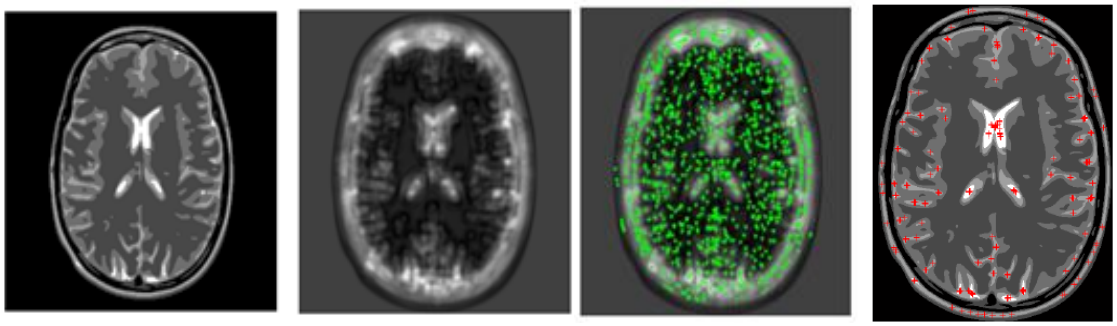
Fig. 4.22: Filtro de Gabor en 2D con los siguientes parámetros: $f = 0.25, \eta = \sqrt{2}, \gamma = \sqrt{2}$ y un kernel de 39×39. **(a)** Filtro con valores absolutos en $\theta = 90°$. **(b)** Filtro con valores absolutos en $\theta = 0°$.

Una vez aplicado el banco de filtros de Gabor sobre la imagen en el espacio de escalas descrito junto con las orientaciones, se procede a identificar los puntos máximos locales en cada nivel de la pirámide. Para esto se utiliza la ecuación de Harris [16] definida como: $det(A) - k * trace^2(A)$, donde A es un tensor calculado como se muestra en la **Ec. 4.22.**

$$A = \begin{bmatrix} L'^2_x(x,y,\sigma_D) & L'_{xy}(x,y,\sigma_D) \\ L'_{xy}(x,y,\sigma_D) & L'^2_y(x,y,\sigma_D) \end{bmatrix} \tag{4.22}$$

donde $\sigma_D = 0.7\sigma, L'_x = gb(x,y,\sigma,0^\circ) * I(x,y)$, $L'_y = gb(x,y,\sigma,90^\circ) * I(x,y)$ y $L'_{xy} = L_x \cdot L_y$.

Del conjunto de puntos máximos encontrados mediante el tensor A, se seleccionan aquellos con un valor de umbral superior al 0.1 del máximo de cada nivel. Finalmente, mediante el método de supresión de no máximos [80] se eligen los puntos más representativos por cada región de la imagen. De esta manera, las regiones son distinguibles. En la **Fig. 4.23** se resume este proceso aplicado en una imagen de resonancia magnética MRI de cerebro.



(a) **(b)** **(c)** **(d)**

Fig. 4.23: Fantoma MRI de cerebro utilizada para identificación de puntos distintivos. **(a)** Imagen original. **(b)** Imagen resultante a la salida del banco de filtros de Gabor. **(c)** Puntos máximos encontrados mediante la ecuación de Harris con umbral de 0.1. **(d)** Puntos distintivos seleccionados mediante supresión de no máximos.

4.6.1.2 Selección automática de escala

La selección automática de la escala de cada característica está basada en el factor del Laplaciano normalizado descrito por K. Mikolajczyk en [81]. A través del espacio de escalas obtenido se busca el máximo valor del Laplaciano entre los niveles σ.

Como se mencionó en la Sección 4.4.2.1, este análisis es posible por la conclusión de Lindeberg, referente a que la amplitud de las derivadas espaciales decrementa cuando el valor de escala aumenta. El Laplaciano de Gaussiano (LoG) es la segunda derivada en el espacio de escalas $L(x,\sigma)$ definido por la componente Gaussiana del filtro de Gabor. Esto se muestra en la **Ec. 4.23.**

$$LoG = \left|\sigma^2 \left(L_{xx}(x,\sigma) + L_{yy}(x,\sigma)\right)\right| \tag{4.23}$$

La búsqueda de los valores máximos del factor LoG se realiza en un espacio 3D donde los dos primeros ejes están relacionados con el tamaño de la imagen y el tercero con el número de niveles de la pirámide correspondiente al espacio de escala (**Fig. 4.24**).

Para que un valor σ se asocie a cada punto distintivo, se recorre la pirámide de escalas comparando si el valor LoG del nivel actual es máximo respecto a los niveles superior e inferior. La región circular final tiene como centro los puntos distintivos con una dimensión igual a 3σ [81].

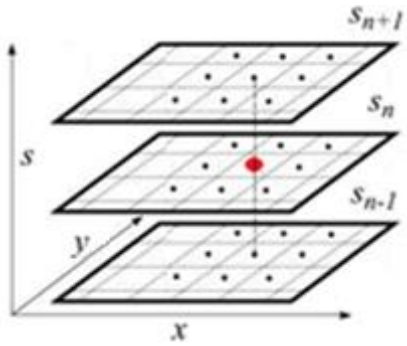
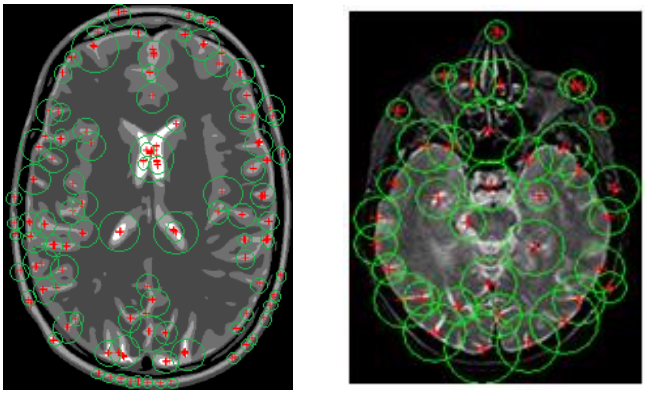


Fig. 4.24: Pirámide LoG en el espacio de escalas $\sigma_{1\ldots m} = k^{1\ldots m}\sigma_0$. El valor de escala se asigna al punto distintivo cuando el valor LoG en el nivel actual, comparado con los niveles superior e inferior, es máximo. Los ejes x, y corresponden a la dimensión de la imagen y el eje s al espacio de escalas.

La **Fig. 4.25** muestra el resultado de este proceso aplicado en dos imágenes distintas.



(a) **(b)**

Fig. 4.25: Selección automática del valor de la escala a cada punto distintivo de acuerdo con el factor LoG en la pirámide del espacio de escalas $L(x,\sigma)$ definido mediante el filtro de Gabor. **(a)** Fantoma MRI de cerebro con dimensiones 512×512. **(b)** Imagen MRI de cerebro en secuencia T2 con tamaño 256×256.

4.6.1.3 Cálculo de los descriptores

Finalmente, para que los puntos distintivos encontrados junto con sus regiones sean considerados *marcadores anatómicos* deben ser descritos mediante un descriptor de textura. Para esto, se utilizaron los momentos invariantes de pseudo-Zernike (PZM) que mapean la información de la imagen en polinomios complejos. Estos momentos son una variante de los momentos de Zernike presentados en la **Sección 4.4.3.3**.

Los momentos de pseudo-Zernike han sido usados porque son más robustos frente al ruido y muestran una habilidad mayor para la descripción de texturas comparada con los momentos de Zernike [82], [83]. El orden del descriptor usado es $n = 7$ y repetición $m = 7$. El cálculo de los momentos PZM están basados en los siguientes parámetros:

- Polinomios radiales:

$$R_{n,m}(r) = \sum_{s=0}^{n-|m|}(-1)^s \frac{(2n-s+1)!}{s!(n-|m|-s)!(n+|m|+1-s)}\, r^{(n-s)} \qquad (4.24)$$

  donde n es el orden del polinomio en el rango $[1,2,\ldots\infty]$ y m son las repeticiones o frecuencia angular con valores positivos dependiendo del grado n donde $|m| \leq n$.

- Funciones base de Zernike:
  $V_{n,m}(x,y) = R_{n,m}(r)e^{jm\theta}$ (4.25)

  donde $r = \sqrt{x^2 + y^2}$ es la longitud del vector desde el origen de los píxeles (x, y) y $\theta = tan^{-1}(y/x)$ es el ángulo entre r y el eje x.

- Momentos PZM sobre las funciones base:
  $PZM_{n,m}\big(f(x,y)\big) = \frac{(n+1)}{\pi\lambda(N)} \sum_{p=0}^{N-1} \sum_{q=0}^{N-1} V_{n,m}^{*}(x,y)\, f(x,y)$ (4.26)

  donde $\lambda(N)$ es la normalización del factor definido como $\lambda(N) = \frac{N^2}{2}$ en [82], $V_{n,m}^{*}(x,y)$ es el complejo conjugado de $V_{n,m}(x,y)$.

Las regiones encontradas mediante la selección automática de escala son descritas mediante los momentos PZM. Esto se muestra en la **Fig. 4.26**.

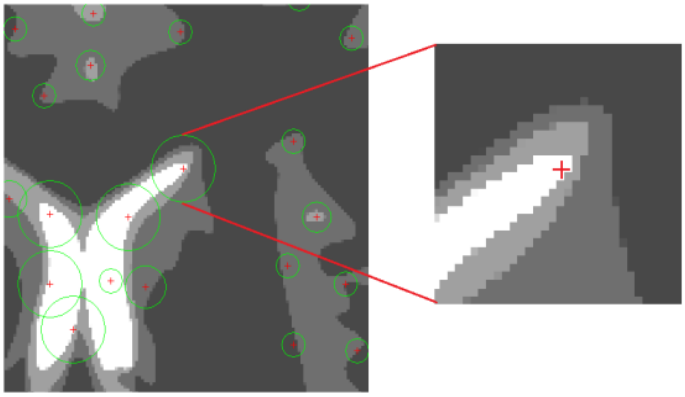


Fig. 4.26: Cada una de las regiones circulares obtenidas mediante los procesos anteriores serán descritas mediante los momentos PZM. El tamaño de las regiones es de $2R \times 2R$.

4.6.1.4 Emparejamiento de descriptores

Como el extractor de marcadores anatómicos propuesto será aplicado para la registración de imágenes se requiere que el emparejamiento, de la información entre dos imágenes base conocidas como imagen de referencia e imagen móvil, sea robusto. Esta extracción generará dos vectores $M_{REF} = [D,\ N_1]$ y $M_{MOV} = [D,\ N_2]$, donde D es el dominio de las imágenes $D = 2$, N_1 y N_2 denota la cantidad de marcadores encontrados en la imagen de referencia y en la móvil.

Para el emparejamiento de los marcadores se aplica primero una normalización, debido a que las regiones que se corresponden entre las dos imágenes analizadas pueden tener distinto tamaño debido a que la imagen móvil va a estar expuesta a deformaciones, en general, de tipo lineal. La normalización de la **Ec. 4.27,** presentada por Zhang et al. en [84] mejora la precisión del emparejamiento.

$$PZM_{n,m}\big(f(x,y)\big) = \left[\frac{PZM_{0,0}}{\pi R^2},\quad \frac{PZM_{1,0}}{PZM_{0,0}}, \frac{PZM_{1,1}}{PZM_{0,0}}, \ldots, \frac{PZM_{7,7}}{PZM_{0,0}}\right] \quad (4.27)$$

Para el emparejamiento se utiliza la métrica de la suma de diferencias cuadráticas *(SSD)*, donde se utiliza un umbral menor a 0.8 para considerar que el emparejamiento es correcto. Esto se detalla en la **Sección 4.4.4**.

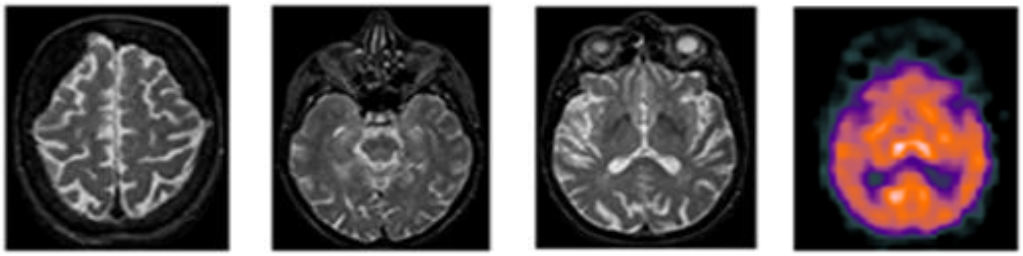
Para medir el rendimiento del algoritmo propuesto se utiliza el factor de repetibilidad, el cual, es usado para medir el rendimiento de algoritmos con este enfoque [41]. La repetibilidad se calcula mediante la **Ec. 4.28.**

$$Rep = \frac{N_T}{\min(N_1,\ N_2)} \tag{4.28}$$

donde N_T denota la cantidad de marcadores emparejados y $\min(N_1,\ N_2)$ es el mínimo valor entre los marcadores encontrados en cada una de las imágenes. La repetibilidad está en el rango $[0, 1]$, donde $R = 1$ indica que existe un emparejamiento del 100% de los marcadores encontrados.

4.6.2 Pruebas realizadas del extractor propuesto

Para analizar el rendimiento del algoritmo propuesto, se ejecutó un grupo de pruebas utilizando imágenes MRI cerebrales con las mismas características a las utilizadas en el **Capítulo 3**. En la **Fig. 4.27**, se muestran las imágenes seleccionadas para este análisis.



(a) **(b)** **(c)** **(d)**

Fig. 4.27: Imágenes de MRI cerebrales facilitadas por el Instituto Radiológico de Mar del Plata. **(a)** MRI en secuencia T1. **(b)** MRI en secuencia T2. **(c)** MRI en secuencia T2. **(d)** MRI en modalidad SPECT.

Los estudios cerebrales se realizan generalmente en un ambiente controlado, por lo que las deformaciones pueden ser mínimas; sin embargo, las imágenes de la **Fig. 4.27** han sido expuestas a un banco de deformaciones rígidas para analizar el rendimiento y la capacidad del algoritmo de emparejar características frente a cambios bruscos en los parámetros de escala, rotación y traslación. Los valores utilizados para estas deformaciones se muestran en la **Tabla 4.1.**

Tabla 4.1: Parámetros aplicados para generar deformaciones rígidas.

Parámetro	**Rango**	**Valor de paso en el rango**
Escala	$[0.7, 1.3]$	0.5
Rotación	$[-30^{\circ}, 30^{\circ}]$	1.5
Traslación	$[-30, 30]$	5

Durante las pruebas, la imagen móvil se genera iterativamente y en forma anidada por cada parámetro de escala, rotación y traslación. La interpolación usada para las transformaciones geométricas es la bilineal debido a su adecuado rendimiento.

Resultados obtenidos

Los resultados que se muestran a continuación han sido agrupados con base en el valor de escala, obteniendo un promedio de la repetibilidad calculada durante los rangos de rotación y traslación. En la **Fig. 4.28**, se muestra el promedio de la repetibilidad por cada una de las imágenes analizadas.

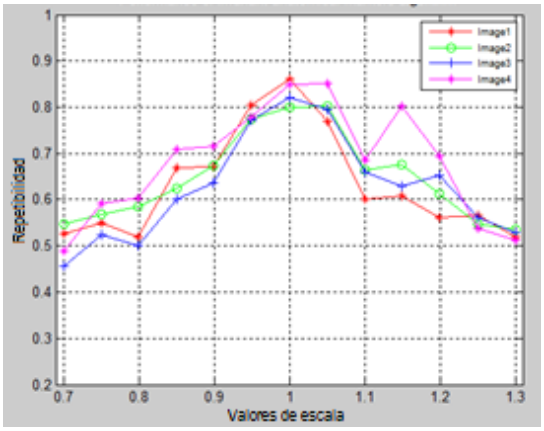
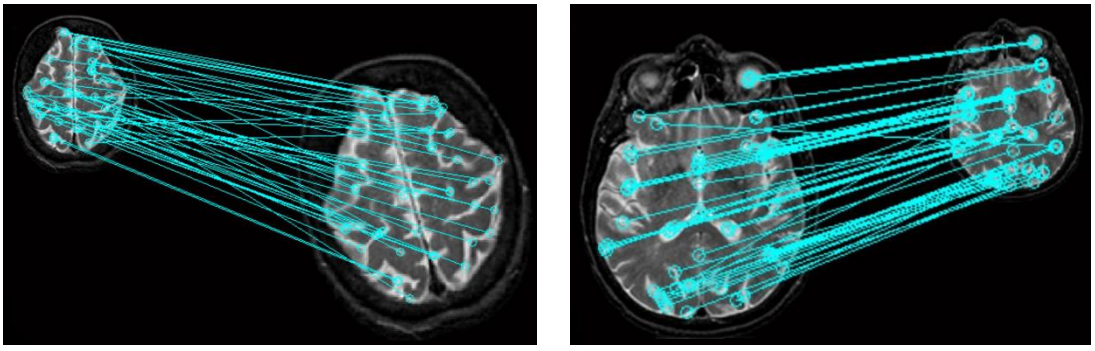


Fig. 4.28: Promedio de repetibilidad obtenido durante el banco de pruebas realizado. El algoritmo muestra un rendimiento adecuado puesto que los valores obtenido están en el rango de $[0.5, 0.85]$.

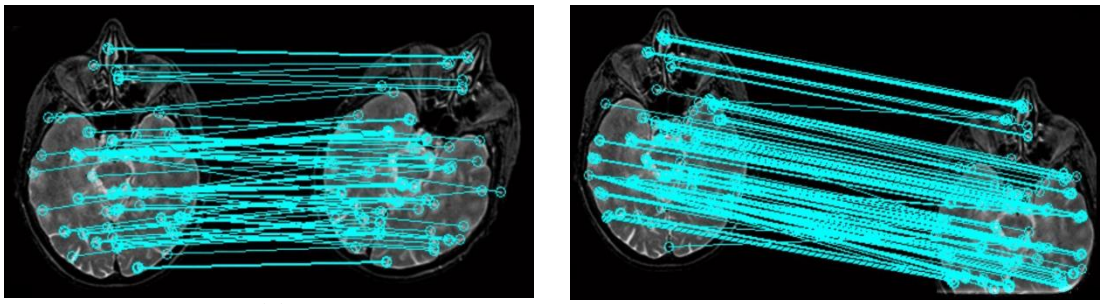
El promedio de repetibilidad obtenido el algoritmo propuesto guarda relación con los resultados obtenidos por Wang et al. en [85], donde se usa el filtro de Gabor con otro enfoque para la detección de esquinas. En dicho trabajo también se muestra la repetibilidad como medida para medir el rendimiento del algoritmo propuesto.

En las **Fig. 4.29 y 4.30**, se muestra la aplicación del algoritmo propuesto para la extracción y emparejamiento de marcadores anatómicos, con valores específicos en los parámetros de deformación presentados de la siguiente manera $[\alpha, \theta, Tx, Ty]$, referidos al valor de escala, rotación, traslación en el eje X y traslación en el eje Y.



(a) **(b)**

Fig. 4.29: Aplicación del algoritmo propuesto frente a cambios grandes de escala. **(a)** MRI en secuencia T1 con imagen móvil deformada con los parámetros $[2.0, 15^{\circ}, 0, 0]$. **(b)** MRI en secuencia T2 con imagen móvil deformada con los parámetros $[0.5, -5^{\circ}, 5, 2]$.



(a) **(b)**

Fig. 4.30: Aplicación del algoritmo propuesto frente a grandes desplazamientos y oclusiones. **(a)** MRI en secuencia T2 con imagen móvil deformada con los parámetros $[1.0, -30^{\circ}, -5, 10]$. **(b)** MRI en secuencia T2 con imagen móvil deformada con los parámetros $[1.0, 5^{\circ}, -60, -50]$.

Conclusiones a partir del análisis de los resultados

Los resultados obtenidos muestran que el detector propuesto alcanza buenos valores en relación al factor de repetibilidad frente a las transformaciones de similaridad. Con lo cual, se puede concluir que el filtro de Gabor es robusto y permite extraer las regiones distintivas de una imagen médica. Además, permite establecer el espacio de escalas necesario para lograr la invariancia de las características (features).

Los momentos PZM al ser normalizados mediante la **Ec. 4.27** permiten tener mayor presión para el emparejamiento de los marcadores anatómicos entre dos imágenes. Este es un aporte realizado para mejorar el rendimiento de estos descriptores. Finalmente, el descriptor propuesto presenta un buen rendimiento frente a grandes cambios (deformaciones) y a posibles oclusiones. Esto permitirá mejorar los resultados durante el proceso de registración, los cuales, son evaluados en la siguiente sección.

4.6.3 Registración de imágenes mediante el extractor propuesto

Para la registración se requiere calcular la mejor matriz de transformación T. En este caso específico la transformación geométrica es la de similaridad (**Sección 2.5.2**). Para el cálculo de los parámetros, mediante el algoritmo iterativo RANSAC, es necesario seleccionar 2 pares de puntos emparejados aleatoriamente.

Al conjunto de puntos emparejados lo llamamos C_{emp} y tiene una dimensión $C_{emp} = [N_T\,,\ 2]$, donde N_T es el total de los marcadores emparejados y 2 columnas para indicar el índice de los marcadores tanto en la imagen de referencia como en la móvil. El cálculo del valor de la escala se realiza mediante la **Ec. 4.29.**

$$s = \frac{\|M_{mov}(P1) - M_{mov}(P2)\|}{\|M_{ref}(P1) - M_{ref}(P2)\|} \tag{4.29}$$

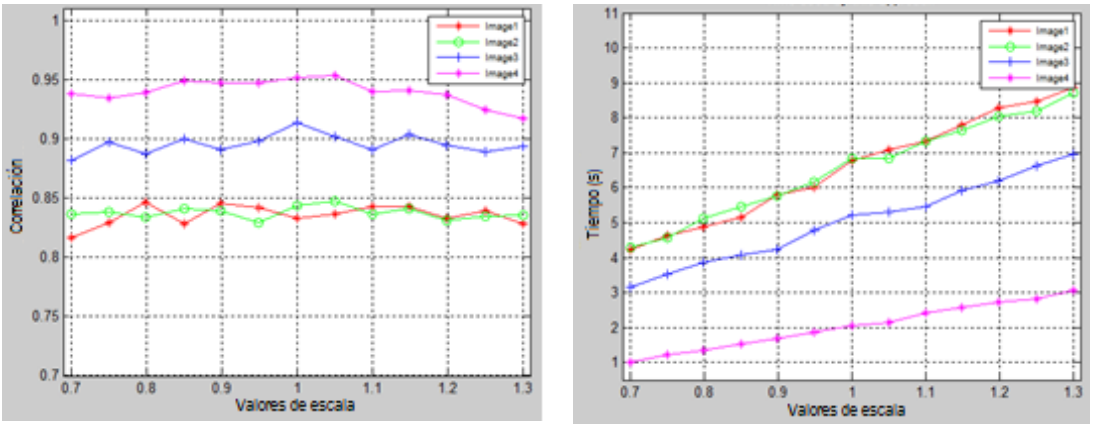
donde M_{mov} corresponde al marcador de la imagen móvil, M_{ref} al marcador de la imagen de referencia, $P1$ y $P2$ son los índices de los puntos seleccionados aleatoriamente. La métrica que se utiliza es la distancia Euclídea.

El ángulo de rotación se calcula mediante $\theta = tan^{-1}(\frac{sin\,\theta}{cos\,\theta})$, donde las funciones $sin(\theta)$ y $cos(\theta)$ se determinan mediante un sistema lineal de dos ecuaciones y dos incógnitas. Finalmente, los valores de traslación se calculan mediante la diferencia de la posición dentro de la imagen de los marcadores seleccionados $P1$ y $P2$ con respecto a los marcadores transformados cuando ya se ha aplicado el valor de escala y rotación.

Este es un proceso de optimización que tiene como función objetivo la minimización de la distancia geométrica del conjunto total de los marcadores emparejados con los transformados, o en su defecto, la maximización de la medida de similaridad entre las imágenes analizadas. En este caso se utiliza el factor de correlación de Pearson, puesto que, la registración es monomodal.

Resultados obtenidos

En la **Fig. 4.31**, se muestra los resultados obtenidos al realizar la registración monomodal basados en la medida de similaridad. Además, se presenta el tiempo de procesamiento utilizado por el algoritmo propuesto durante la extracción de marcadores anatómicos y la registración. De forma similar, que, en la sección anterior, los resultados muestran el promedio obtenido agrupados de acuerdo con el valor de escala.



(a) **(b)**

Fig. 4.31: Resultados obtenidos en la registración utilizando el banco de pruebas propuesto. **(a)** Promedio de correlación de Pearson obtenida por las 4 imágenes. **(b)** Promedio de tiempo de procesamiento (en segundos) utilizado por el algoritmo propuesto.

Conclusiones a partir del análisis de los resultados

Los resultados obtenidos muestran un rendimiento acorde al esperado para el algoritmo propuesto. El valor de correlación se encuentra en el rango $[0.82 - 0.95]$, por tanto, el error final de la imagen registrada respecto de la imagen de referencia es mínimo.

El tiempo utilizado varía de acuerdo con el incremento de la escala de las imágenes. Esto tiene relación con el aumento de la cantidad de puntos distintivos y posteriores marcadores anatómicos, ya que la cantidad de información procesada es mayor.

Sin embargo, el tiempo máximo utilizado está en el orden de $[10 - 12]$ segundos, el cual, podría ser considerado óptimo para tareas que no requieren procesamiento en tiempo real, como es el caso de la registración de imágenes.

Resultados obtenidos comparando con SIFT y SURF

Finalmente, se realizó una comparación del rendimiento, con dos algoritmos muy utilizados para la extracción de características en imágenes, como son: SIFT [29] y SURF [30]. Estos algoritmos han sido expuestos al mismo banco de pruebas propuesto. Los resultados obtenidos se muestran a continuación.

En la **Fig. 4.32**, se muestra la comparación del promedio del factor de repetibilidad de las imágenes utilizadas agrupadas de acuerdo con el valor de escala. En la **Fig. 4.33**, se muestran los promedios de la correlación y el tiempo de procesamiento obtenidos por cada algoritmo.

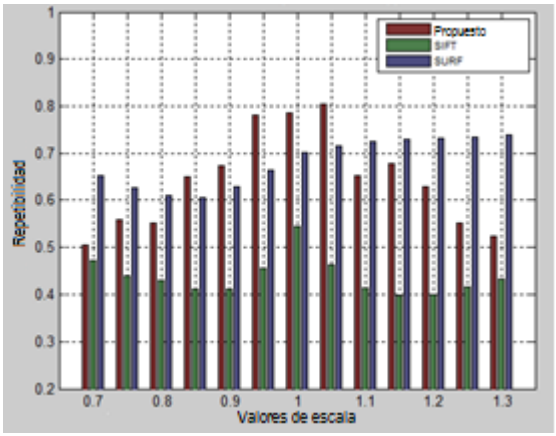
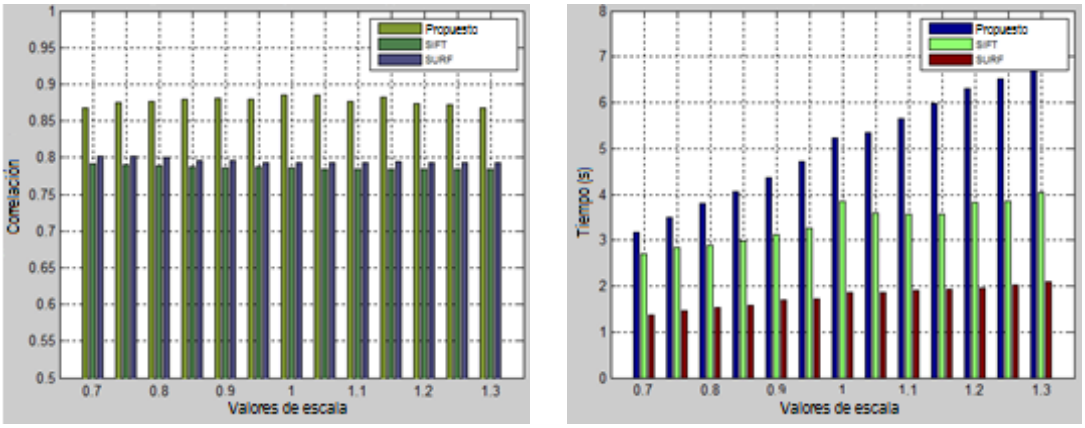


Fig. 4.32: Comparación del rendimiento durante la extracción de características mediante el promedio del factor de repetibilidad. Este promedio es con base en las 4 imágenes utilizadas en las pruebas y agrupadas de acuerdo con el valor de escala.



(a) **(b)**

Fig. 4.33: Resultados obtenidos durante la registración de imágenes. **(a)** Promedio del factor de correlación obtenido por los 3 algoritmos en las 4 imágenes utilizadas. **(b)** Promedio del tiempo de procesamiento utilizado por los 3 algoritmos.

Conclusiones a partir del análisis de los resultados

El algoritmo propuesto muestra resultados interesantes al compararlo con SIFT y SURF, los cuales, son algoritmos robustos y utilizados en diversas aplicaciones. El promedio del factor de repetibilidad del algoritmo propuesto es cercano y en varios casos superior al de SIFT y SURF en el espacio de escala analizado. Esto muestra que el rendimiento del algoritmo es prometedor para la extracción y emparejamiento de características o marcadores anatómicos.

En el caso de la registración, el algoritmo propuesto supera en todos los casos a los resultados obtenidos por los otros dos algoritmos. El promedio de estos valores de la medida de similaridad está dentro del rango $[0.80 - 0.89]$. Este resultado es el que más relevancia tiene, ya que, la registración es el objetivo principal del presente trabajo.

Respecto del tiempo de procesamiento, el algoritmo propuesto utiliza en promedio de $[8 - 10]$ segundos en el valor máximo de escala. Sin embargo, comparado con los otros algoritmos es el que mayor tiempo utiliza. El algoritmo que optimiza el tiempo es SURF. Sin embargo, el tiempo para aplicaciones que no requieran procesamiento en tiempo real, sigue siendo aceptable.

Referencias

[1] K. Grauman and B. Leibe, *Visual Object Recognition*. 2010.

[2] J. Rongrong, H. Yao, Y. Gao, L.-Y. Duan, and Q. Dai, "Introduction," in *Learning-based local visual representation and indexing*, 2015, pp. 1–15.

[3] T. Tuytelaars and K. Mikolajczyk, "Local Invariant Feature Detectors: A Survey," *Found. Trends® Comput. Graph. Vis.*, vol. 3, no. 3, pp. 177–280, 2007.

[4] N. M. Zaitoun and M. J. Aqel, "Survey on Image Segmentation Techniques," *Procedia Comput. Sci.*, vol. 65, pp. 797–806, Jan. 2015.

[5] M. S. Nixon and A. S. Aguado, *Feature extraction & image processing for computer vision*, 3rd ed. Academic Press, 2012.

[6] L. Van Gool, T. Tuytelaars, and A. Turina, "Local Features for Image Retrieval," Springer, Dordrecht, 2001, pp. 21–41.

[7] B. Heisele and C. Rocha, "Local shape features for object recognition," in *2008 19th International Conference on Pattern Recognition*, 2008, pp. 1–4.

[8] D. G. Lowe, "Local feature view clustering for 3D object recognition," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 1, p. I-682-I-688.

[9] X. Meng, Z. Wang, and L. Wu, "Building global image features for scene recognition," *Pattern Recognit.*, vol. 45, no. 1, pp. 373–380, Jan. 2012.

[10] A. M. Patel, "A Survey on Object Based Image Retrieval using Local and Global Features," 2014.

[11] K. Murphy, A. Torralba, D. Eaton, and W. Freeman, "Object Detection and Localization Using Local and Global Features," Springer, Berlin, Heidelberg, 2006, pp. 382–400.

[12] C. Carson, S. Belongie, H. Greenspan, and J. Malik, "Blobworld: image segmentation using expectation-maximization and its application to image querying," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 8, pp. 1026–1038, Aug. 2002.

[13] R. Sicre and T. Gevers, "DENSE sampling of features for image retrieval," in *2014 IEEE International Conference on Image Processing (ICIP)*, 2014, pp. 3057–3061.

[14] E. Nowak, F. Jurie, and B. Triggs, "Sampling Strategies for Bag-of-Features Image Classification," Springer, Berlin, Heidelberg, 2006, pp. 490–503.

[15] H. P. Moravec, "Obstacle Avoidance and Navigatio n in the Real World by a Seeing Robot Rover," Carnegie-Mellon University, 1980.

[16] C. Harris and M. Stephens, "A combined corner and edge detector," in *Alvey Vision Conference*, 1988, pp. 147–151.

[17] Förstner W. and E. Gülch, "A Fast Operator for Detection and Precise Location of Distinct Points, Corners and Centres of Circular Features," *Proc. ISPRS intercommission Conf. fast Process. Photogramm. data*, pp. 281–305, 1987.

[18] S. M. Smith and J. M. Brady, "SUSAN A New Approach to Low Level Image Processing," *Int. J. Comput. Vis.*, vol. 23, no. 34, pp. 45–78, 1997.

[19] Z. Zhang, R. Deriche, O. Faugeras, Q.-T. Luong, and Q.-T. A. Luong, "A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry," *Artif. Intell.*, vol. 78, pp. 87–119, 1995.

[20] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. 2003.

[21] C. Schmid and R. Mohr, "Local Grayvalue Invariants for Image Retrieval," *IEEE Trans-actions Pattern Anal. Mach. Intell. Inst. Electr. Electron. En-gineers*, vol. 19, no. 5, pp. 530–534, 1997.

[22] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, 1999, pp. 1150–1157 vol.2.

[23] K. Mikolajczyk and C. Schmid, "Indexing based on scale invariant interest points," in *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, 2001, vol. 1, pp. 525–531.

[24] T. Lindeberg, "Feature Detection with Automatic Scale Selection," *Int. J. Comput. Vis.*, vol. 30, no. 2, pp. 79–116, 1996.

[25] T. Lindeberg, "Scale Selection," *Comput. Vis. A Ref. Guid.*, pp. 701–713, 2014.

[26] R. A. Young, "The Gaussian Derivative Theory Of Spatial Vision: Analysis Of Cortical Cell Receptive Field Line-Weighting Profiles," Warren, Michigan, 1985.

[27] J. Zhang, Q. Chen, X. Bai, Q. Sun, H. Sun, and D. Xia, "An Advanced Harris-Laplace Feature Detector with High Repeatability," in *2009 2nd International Congress on Image and Signal Processing*, 2009, pp. 1–5.

[28] X. Xu, "Blob Detection with the Determinant of the Hessian," *Li S., Liu C., Wang Y. Pattern Recognition. CCPR 2014. Commun. Comput. Inf. Sci.*, vol. 483, pp. 72–80, 2014.

[29] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[30] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-Up Robust Features (SURF)," *Comput. Vis. Image Underst.*, vol. 110, no. 3, pp. 346–359, 2008.

[31] E. Rosten and T. Drummond, "Fusing Points and Lines for High Performance Tracking," *Proc. Int. Conf. Comput. Vis.*, pp. 1508–1511, 2005.

[32] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," *Proc. Eur. Conf. Comput. Vis.*, pp. 430–443, 2006.

[33] T. Tuytelaars and L. Van Gool, "Wide Baseline Stereo Matching based on Local, Affinely Invariant Regions," *Proc. Br. Mach. Vis. Conf.*, pp. 412–425, 2000.

[34] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust Wide Baseline Stereo from Maximally Stable Extremal Regions," *Proc. Br. Mach. Vis. Conf.*, pp. 384–393, 2002.

[35] G. Mori, Xiaofeng Ren, A. A. Efros, and J. Malik, "Recovering human body con .gurations:combining segmentation and recognition," in *Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, vol. 2, pp. 326–333.

[36] T. Lindeberg, "Scale-space theory: A basic tool for analysing structures at different scales," *Appl. Stat.*, vol. 21, no. 2, pp. 225–270, 1994.

[37] R. Klette, "Image Processing," in *Concise Computer Vision*, Springer-Verlag, Ed. London, pp. 43–87, 2014.

[38] E. H. Adelson, C. H. Anderson, | J R Bergen, | P J Burt, and | J M Ogden, "Pyramid methods in image processing," *RCA Eng.*, vol. 29, no. 6, pp. 33–41, 1984.

[39] J. L. Crowley and O. Riff, "Fast Computation of Scale Normalised Gaussian Receptive Fields," in *Scale Space Methods in Computer Vision* , Springer, Berlin, Heidelberg, 2003, pp. 584–598.

[40] E. P. Simoncelli and W. T. Freeman, "The steerable pyramid: a flexible architecture for multi-scale derivative computation," in *Proceedings., International Conference on Image Processing*, vol. 3, pp. 444–447, 1995.

[41] A. Ardeshir Goshtasby, *Image Registration: Principles, Tools and Methods (Advances in Computer Vision and Pattern Recognition)*, Springer. 2012.

[42] K. Mikolajczyk and C. Schmid, "A Performance Evaluation of Local Descriptors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 10, pp. 1615–1630, 2005.

[43] S. A. J. Winder and M. Brown, "Learning Local Image Descriptors," *Comput. Vis. Pattern Recognit.*, pp. 1–8, 2007.

[44] A. M. Ferman, A. M. Tekalp, and R. Mehrotra, "Robust color histogram descriptors for video segment retrieval and identification," *IEEE Trans. Image Process.*, vol. 11, no. 5, pp. 497–508, May 2002.

[45] M. Yasmin, M. Sharif, lsma lrum, and S. Mohsin, "Powerful Descriptor for Image Retrieval Based on Angle Edge and Histograms," *J. Appl. Res. Technol.*, vol. 11, no. 5, pp. 727–732, Oct. 2013.

[46] N. Prajapati, A. Kumar Nandanwar, and G. S. Prajapati, "Edge Histogram Descriptor, Geometric Moment and Sobel Edge Detector Combined Features Based Object Recognition and Retrieval System," *Int. J. Comput. Sci. Inf. Technol.*, vol. 7, no. 1, pp. 407–412, 2016.

[47] O. Déniz, G. Bueno, J. Salido, and F. De la Torre, "Face recognition using Histograms of Oriented Gradients," *Pattern Recognit. Lett.*, vol. 32, no. 12, pp. 1598–1603, Sep. 2011.

[48] B. Başa, "Implementation of Hog Edge Detection Algorithm Onfpga's," *Procedia - Soc. Behav. Sci.*, vol. 174, pp. 1567–1575, Feb. 2015.

[49] Zhihu Huang and Jinsong Leng, "Analysis of Hu's moment invariants on image scaling and rotation," in *2010 2nd International Conference on Computer Engineering and Technology*, 2010, pp. V7-476-V7-480.

[50] Ming-Kuei Hu, "Visual pattern recognition by moment invariants," *IEEE Trans. Inf. Theory*, vol. 8, no. 2, pp. 179–187, Feb. 1962.

[51] B.-C. Li, "A new computation of geometric moments," *Pattern Recognit.*, vol. 26, no. 1, pp. 109–113, Jan. 1993.

[52] M. Arafah and Q. A. Moghli, "Efficient Image Recognition Technique Using Invariant Moments and Principle Component Analysis," *J. Data Anal. Inf. Process.*, vol. 5, pp. 1–10, 2017.

[53] E. M. Arvacheh and H. R. Tizhoosh, "Pattern Analysis Using Zernike Moments," in *2005 IEEE Instrumentationand Measurement Technology Conference Proceedings*, 2005, vol. 2, pp. 1574–1578.

[54] P. Ananth raj and A. Venkataramana, "Radial Krawtchouk moments for rotational invariant pattern recognition," in *2007 6th International Conference on Information, Communications & Signal Processing*, 2007, pp. 1–5.

[55] J. B. F. P. Crespo and P. M. Q. Aguiar, "Revisiting Complex Moments For 2D Shape Representation and Image Normalization," *IEEE Trans. IMAGE Process.*, vol. 20, no. 10, pp. 2896–2911, 2010.

[56] S.-K. Hwang and W.-Y. Kim, "A novel approach to the fast computation of Zernike moments," *Pattern Recognit.*, vol. 39, no. 11, pp. 2065–2076, Nov. 2006.

[57] M. Yang *et al.*, "A survey of Shape Feature Extraction Techniques.," in *Pattern Recognition Techniques, Technology and Applications*, InTech, 2008, pp. 43–90.

[58] S. X. Liao and M. Pawlak, "On the accuracy of Zernike moments for image analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 12, pp. 1358–1364, 1998.

[59] G. A. Papakostas, E. G. Karakasis, and D. E. Koulouriotis, "Orthogonal Image Moment Invariants," in *Cross-Disciplinary Applications of Artificial Intelligence and Pattern Recognition: Advancing Technologies*, IGI Global, 2013, pp. 15–32.

[60] H. Huibao Lin, J. Si, and G. P. Abousleman, "Orthogonal Rotation-Invariant Moments for Digital Image Processing," *IEEE Trans. Image Process.*, vol. 17, no. 3, pp. 272–282, Mar. 2008.

[61] T. Suk and J. Flusser, "Affine moment invariants generated by graph method," *Pattern Recognit.*, vol. 44, no. 9, pp. 2047–2056, Sep. 2011.

[62] H. R. Kanan and S. Salkhordeh, "Rotation invariant multi-frame image super resolution reconstruction using Pseudo Zernike Moments," *Signal Processing*, vol. 118, pp. 103–114, Jan. 2016.

[63] B. Honarvar, R. Paramesran, and C.-L. Lim, "Image reconstruction from a complete set of geometric and complex moments," *Signal Processing*, vol. 98, pp. 224–232, 2014.

[64] J. Sarvaiya, S. Patnaik, and H. Goklani, "Image Registration Using Mexican-Hat Wavelets and Invariant Moments," Springer, Berlin, Heidelberg, 2011, pp. 574–577.

[65] M. Favorskaya, D. Pyankov, and A. Popov, "Motion estimations based on invariant moments for frames interpolation in stereovision," *Procedia Comput. Sci.*, vol. 22, pp. 1102–1111, 2013.

[66] M. B. Hisham, S. N. Yaakob, R. A. A. Raof, A. . A. Nazren, and N. M. W. Embedded, "Template Matching using Sum of Squared Difference and Normalized Cross Correlation," in *2015 IEEE Student Conference on Research and Development (SCOReD)*, pp. 100–104, 2005.

[67] Z. Zhao *et al.*, "Improved Direct Linear Transformation for Parameter Decoupling in Camera Calibration," *Algorithms*, vol. 9, no. 2, p. 31, Apr. 2016.

[68] Lloyd N. Trefethen and D. Bau, *Numerical linear algebra*, 1st ed. Society for Industrial and Applied Mathematics, 1997.

[69] J. D. Foley, M. A. Fischler, and R. C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Apphcatlons to Image Analysis and Automated Cartography," *Graph. Image Process.*, vol. 24, pp. 381–395, 1981.

[70] B. Vikhoff-Baaz, A. C. Bergh, G. Starck, S. Ekholm, and C. Wikkelsö, "A new set of fiducial markers for MRI, CT and SPET alignment.," *Nucl. Med. Commun.*, vol. 18, no. 12, pp. 1148–54, Dec. 1997.

[71] J. B. West, J. M. Fitzpatrick, S. A. Toms, C. R. Maurer, and R. J. Maciunas, "Fiducial Point Placement and the Accuracy of Point-based, Rigid Body Registration," *Neurosurgery*, vol. 48, pp. 810–817, 2001.

[72] A. Seginer, "Rigid-body point-based registration: The distribution of the target registration error when the fiducial registration errors are given," *Med. Image Anal.*, vol. 15, no. 4, pp. 397–413, Aug. 2011.

[73] M. Franaszek and G. S. Cheok, "Selection of fiducial locations and performance metrics for point-based rigid-body registration," *Precis. Eng.*, vol. 47, pp. 362–374, Jan. 2017.

[74] Isa-Jara R., Meschino G., Ballarin V., "Identification of invariant anatomical markers for medical image registration using Gabor Filters*",* XXI CONGRESO ARGENTINO DE BIOINGENIERÍA, ISBN: 978-950-33-1406-7. Córdoba, 2017.

[75] J. Ilonen, J.-K. Kämäräinen, H. Kälviäinen, and J.-K. Kamarainen, "Efficient computation of Gabor features," 2005.

[76] J. G. Daugman, "Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters.," *J. Opt. Soc. Am. A.*, vol. 2, no. 7, pp. 1160–9, Jul. 1985.

[77] M. Noruzi, M. Vafadoost, and M. S. Moin, "Iris Recognition: Localization, Segmentation and Feature Extraction Based on Gabor Transform," *Comput. Sci. Its Appl. - ICCSA 2006*, pp. 1180–1189, 2006.

[78] G. Amayeh, A. Tavakkoli, and G. Bebis, "Accurate and Efficient Computation of Gabor Features in Real-Time Applications," *Adv. Vis. Comput. ISVC 2009.*, vol. Vol. 5875, pp. 243, 252, Nov. 2009.

[79] L. El Shafey, R. Wallace, and S. Marcel, "Face Verification using Gabor Filtering and Adapted Gaussian Mixture Models," *Int. Conf. Biometrics Spec. Interes. Gr. (BIOSIG)*, 2012.

[80] A. Neubeck and L. Van Gool, "Efficient Non-Maximum Suppression," in *18th International Conference on Pattern Recognition (ICPR'06)*, pp. 850–855, 2006.

[81] K. Mikolajczyk and C. Schmid, "Scale & Affine Invariant Interest Point Detectors," *Int. J. Comput. Vis.*, vol. 60, no. 1, pp. 63–86, 2004.

[82] C. Singh, E. Walia, and R. Upneja, "Accurate calculation of Zernike moments," *Inf. Sci. (Ny).*, vol. 233, pp. 255–275, Jun. 2013.

[83] H. R. Kanan and S. Salkhordeh, "Rotation invariant multi-frame image super resolution reconstruction using Pseudo Zernike Moments," *Signal Processing*, vol. 118, pp. 103–114, Jan. 2016.

[84] D. Zhang and G. Lu, "Shape-based image retrieval using generic Fourier descriptor," *Signal Process. Image Commun.*, vol. 17, no. 10, pp. 825–848, Nov. 2002.

[85] F.-P. Wang, W.-C. Zhang, Z.-F. Zhou, and L. Zhu, "Corner detection using Gabor filter," *IET Image Process.*, vol. 8, no. 11, pp. 639–646, Nov. 2014.

Capítulo 5 Inteligencia computacional

> Soy de las que piensan que la ciencia tiene una gran belleza. Un científico en su laboratorio no es solo un técnico: es también un niño colocado ante fenómenos naturales que le impresionan como un cuento de hadas.
>
> *Marie Curie*

5.1 Introducción

Durante miles de años, los humanos han tratado de comprender los mecanismos del pensamiento. Este es el rasgo característico que los distingue del resto de seres vivos, el de *ser conscientes*. El desarrollo gradual de la inteligencia[12] ha permitido mejorar las capacidades para percibir, entender, predecir y manipular el mundo que los rodea.

La *inteligencia artificial* (AI) es un campo de investigación multidisciplinar que ha aportado nuevos conocimientos acerca de la inteligencia junto con sus procesos fisiológicos y psicológicos, incluso yendo más allá, con el diseño y construcción de *agentes artificiales inteligentes* [1].

Vinculada a la AI se encuentra la *inteligencia computacional* (CI). Aunque se tiende a confundirlas como sinónimos, existen diferencias entre ellas. Un posible enfoque permitiría considerar a la AI, como se mencionó anteriormente, como multidisciplinar, en la que se combinan conocimientos de dos grandes áreas: la *científica/técnica y* la *humanista.* En la primera están las ciencias computacionales, matemática, biología, física, entre otras. En la segunda se encuentran la psicología, filosofía, sociología, lingüística, entre otras [2].

Por tanto, la CI se convierte en una parte de la AI ya que su principal aporte se realiza en el área científico/técnica, específicamente el campo de las ciencias computacionales. La CI está relacionada con el diseño de *sistemas informáticos inteligentes*. Se dice que un sistema es *inteligente* cuando muestra ciertas características que se asocian tanto al comportamiento humano, para la comprensión del lenguaje, el aprendizaje de razonamiento, la resolución de problemas [3], como al de ciertas especies animales, como las hormigas o las aves durante su proceso de búsqueda de alimento (**Sección 3.6**).

La CI ha captado gran interés en las últimas décadas a nivel académico con un gran impacto en la industria. Actualmente, una cantidad considerable de aplicaciones utilizan varios paradigmas de la CI como la lógica difusa, los algoritmos genéticos, la inteligencia de enjambres y las redes neuronales para la ejecución de sus tareas [4]. Ejemplos de esto se encuentran en el reconocimiento de rostros, segmentación y reconocimiento de imágenes, sistemas de vigilancia automática, diagnóstico médico, entre muchos otros.

[12] Según el diccionario de la Real Academia de la Lengua, inteligencia es la capacidad de entender o comprender. Capacidad de resolver problemas. Conocimiento, comprensión y acto de entender.

5.2 Inteligencia computacional y sus paradigmas

Mediante la CI se modelan algoritmos para resolver problemas complejos. El modelado basado en el comportamiento de agentes inteligentes que se encuentran en la naturaleza ha permitido que la CI alcance gran éxito. Usando la inteligencia humana como referencia, los algoritmos buscan imitar la capacidad de pensar y razonar frente a nuevas situaciones [2].

Por tanto, un agente artificial inteligente debe ser flexible, adaptando su comportamiento a los diversos cambios del ambiente, aprendiendo de la experiencia y adaptándose a las nuevas situaciones para abstraer, generalizar, descubrir y asociar, con la finalidad de realizar buenas elecciones de acuerdo con sus percepciones limitadas y cálculo finito. El objetivo no es que estos agentes reemplacen a la inteligencia humana, sino que la asistan como una herramienta poderosa para mejorar la capacidad de análisis, cálculo y toma de decisiones [3], [5].

Los paradigmas más conocidos dentro de la IC son: las redes neuronales artificiales (ANN) [6], [7], los sistemas difusos (FS) [8], [9], la computación evolutiva (EC) [10], [11], la inteligencia de enjambres (SI) [12], [13] y las máquinas de soporte vectorial (SVM) [14], [15]. Todos estos paradigmas están basados en métodos probabilísticos y la fusión de ellos, en algunos casos, dan lugar a los sistemas híbridos [16], [17].

Dentro del procesamiento de imágenes, estos paradigmas han sido aplicados con distintos propósitos como, por ejemplo, la mejora de imágenes de RM mediante redes neuronales [18], segmentación y detección de bordes mediante sistemas difusos [19], para reducir la dimensionalidad de los datos mediante computación evolutiva [20] y clasificación de imágenes mediante máquinas de soporte vectorial [21].

5.3 Aprendizaje de máquina

Actualmente, la CI ha demostrado una gran habilidad para descubrir patrones y sus dependencias a partir de la extracción de información de un conjunto de datos empíricos usando métodos computacionales y estadísticos. El área específica que se dedica a investigar y mejorar estos procesos se conoce como *Aprendizaje de máquina* o más conocida en inglés como *Machine Learning* [22].

La mayoría de los métodos de aprendizaje de máquina están basados en la inferencia *inductiva-deductiva.* La inferencia *inductiva* permite extraer reglas y patrones de un conjunto de datos. La generalización se realiza partiendo de casos particulares (datos del conjunto) para llegar a establecer nuevos patrones y dependencias (modelo estimado). La inferencia *deductiva* permite aplicar el modelo estimado y aplicar el conocimiento general para predicciones específicas. La generalización se limita a suposiciones definidas *a priori*, por ejemplo, los límites de las variables de entrada [23].

Para que los sistemas computacionales cuenten con la habilidad de la inferencia es necesario disponer de una representación adecuada de los datos. Esta representación se basa en un *espacio de características* y un *modelo estadístico.*

El espacio de características (*feature space*) es una representación matemática de los datos que son el núcleo del algoritmo de aprendizaje. Los patrones encontrados en el espacio de características provienen inevitablemente de modelos estadísticos. Los *modelos estadísticos* realizan proposiciones sobre grupos de datos formados a partir del conjunto de datos disponible. La selección se realiza con base en un muestreo aleatorio [22].

En el estado del arte actual, los enfoques más utilizados para el aprendizaje de máquina están divididos en tres categorías principales: aprendizaje no supervisado, aprendizaje supervisado y aprendizaje por refuerzo (**Fig. 5.1**).

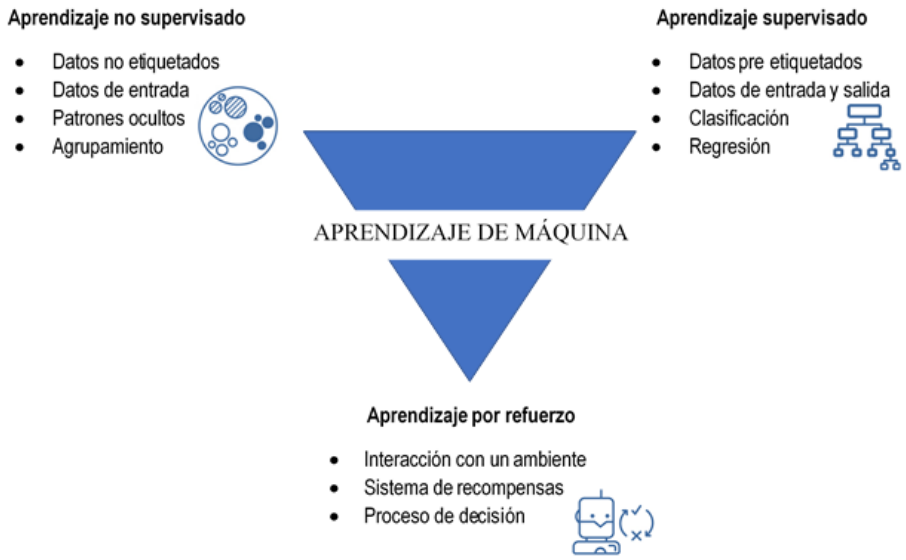


Fig. 5.1: Los tres enfoques principales utilizados en el aprendizaje de máquina. Cada uno de ellos utiliza métodos computacionales y estadísticos en sus procesos con el objetivo de emular el aprendizaje humano.

5.3.1 Aprendizaje no supervisado

Usando este tipo de aprendizaje, los algoritmos modelan el conjunto de datos de entrada buscando organizar la información para encontrar patrones ocultos. Los algoritmos no tienen un objetivo específico que alcanzar, ya que los ejemplos de salida no están disponibles o, en ciertos casos, ni siquiera existen.

Relacionándolo con la forma en la que aprenden los humanos, este tipo de aprendizaje no cuenta con un *supervisor* (objetivos deseados), por tanto, requiere que el *aprendiz* (algoritmo) encuentre una estructura a la que los datos se ajusten (**Fig. 5.2**). Esto se realiza mediante una autoorganización [2], [26]. Este método encuentra aplicaciones en áreas como la bioinformática, reconocimiento de escritura, segmentación de imágenes, reconocimiento de patrones y de voz, recuperación de información, entre otras.

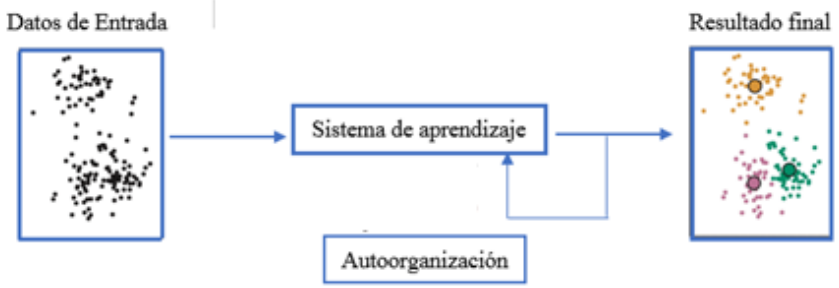


Fig. 5.2: Esquema de los algoritmos que utilizan el aprendizaje no supervisado. La retroalimentación indica que el algoritmo es iterativo hasta alcanzar una convergencia óptima.

Los algoritmos más representativos de este aprendizaje son los que buscan el agrupamiento (*clustering*) de los datos [24], [25] y los mapas autoorganizados (*SOM*) [27], [28], [29].

5.3.2 Aprendizaje supervisado

El aprendizaje supervisado permite mapear datos de entrada a salidas deseadas. En este caso se cuenta con un *supervisor* que proporciona el conocimiento mediante la entrega de las salidas objetivo. A este proceso se lo conoce como *entrenamiento*, durante el cual, el *aprendiz* (algoritmo) ajusta los parámetros del sistema mediante el error obtenido entre la salida obtenida y la salida objetivo (**Fig. 5.3**) [2], [26].

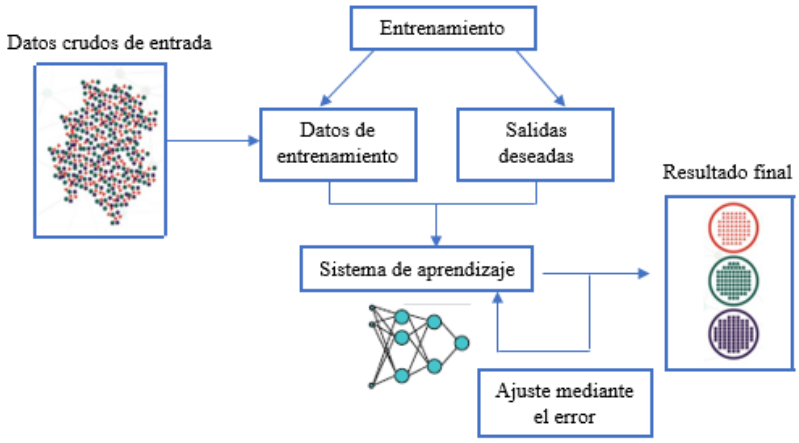


Fig. 5.3: Esquema de los algoritmos que utilizan el aprendizaje supervisado. En la etapa de entrenamiento el sistema ajusta sus parámetros mediante la reducción del error.

Estos sistemas se usan tanto para regresión como para clasificación y es el paradigma utilizado para el entrenamiento de las redes neuronales totalmente conectadas (*feed forward, fully-connected*) y las máquinas de soporte vectorial, entre muchos otros modelos.

5.3.3 Aprendizaje por refuerzo

Estos algoritmos están basados en la idea del aprendizaje mediante la interacción directa con un entorno o ambiente. Para esto se debe adaptar el comportamiento a los cambios que puedan existir, maximizando una función objetivo específica del ambiente. El mecanismo de aprendizaje se basa en acciones de *prueba-error* y la evaluación de la señal de recompensa obtenida. Cada acción tiene un impacto en el ambiente, el cual provee una retroalimentación, en forma de recompensa o penalización, para guiar el aprendizaje del algoritmo (**Fig. 5.4**) [2], [30].

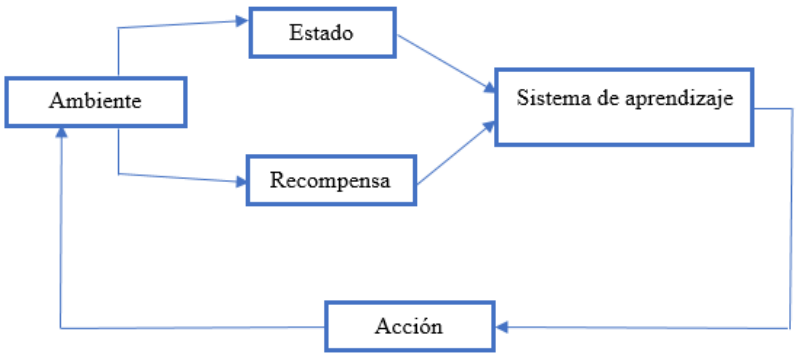


Fig. 5.4: Esquema de los algoritmos que utilizan el aprendizaje por refuerzo. El sistema de aprendizaje influye en el ambiente mediante una acción y el ambiente le guía mediante la recompensa.

El objetivo es encontrar el comportamiento óptimo, es decir, aquel cuyas acciones maximicen la recompensa a largo plazo. Este aprendizaje se utiliza a menudo en agentes inteligentes y robótica. El aprendizaje por refuerzo se abordará en profundidad en la siguiente sección, pues ha sido utilizado para entrenar un agente capaz de aprender a registrar imágenes médicas.

5.3.4 Aprendizaje profundo

El aprendizaje profundo, mejor conocido en inglés como *Deep Learning*, es uno de los enfoques de aprendizaje de máquina con un gran impacto en investigación y desarrollo a partir del año 2015 debido al incremento en la capacidad de procesamiento de las unidades GPU, la disminución relativa en el costo del hardware y los avances en los algoritmos de aprendizaje computacional [31]. Si bien su surgimiento puede parecer nuevo, sus fundamentos y predecesores datan de los años 40. Entre las áreas que han permitido el surgimiento del aprendizaje profundo, se pueden mencionar las siguientes: cibernética, conexionismo, redes neuronales artificiales, modelos lineales y representación distribuida [32].

Como se mencionó anteriormente, el núcleo de todo enfoque de aprendizaje de máquina es la representación de los datos de entrada en un espacio de características (*features*) adecuado. A medida que la complejidad de los problemas aumenta, los algoritmos deben ser capaces de extraer un conjunto de buenas características para obtener un óptimo rendimiento durante el proceso de aprendizaje, lo que le permitirá adaptarse exitosamente a nuevas tareas con una mínima intervención humana. Sin embargo, la complejidad de la extracción de características es alta debido al nivel de abstracción requerido a partir del conjunto de datos de entrada [32].

En esto último es donde el aprendizaje profundo ha sido aplicado con gran éxito. En la introducción de nuevas representaciones a partir de otras *más simples* utilizando una arquitectura jerárquica [31], es decir, permitir que un algoritmo construya conceptos complejos a partir de conceptos simples. Por ejemplo, un algoritmo entrenado bajo este enfoque será capaz de construir un concepto complejo de un *objeto específico* (como una persona, auto, perro, etc.) que está dentro de una imagen utilizando conceptos simples como sus *contornos y esquinas*, los cuales, a su vez están agrupados en otro concepto simple denominado *bordes* [32]. En la **Fig. 5.5** se muestra una red neuronal profunda entrenada para identificación de rostros, con tres capas ocultas dispuestas de manera jerárquica para la extracción de características.

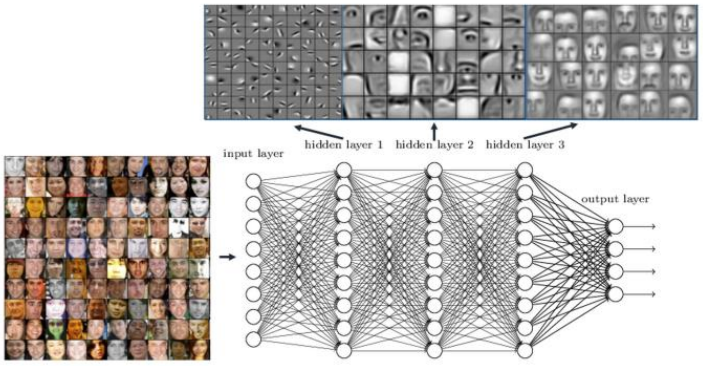


Fig. 5.5: Red neuronal profunda para identificación de rostros con 3 capas ocultas dispuestas jerárquicamente. Capa de entrada para ingreso del conjunto de datos de entrada en crudo. **Capa 1:** extracción de esquinas, contornos, bordes. **Capa 2:** vectores de características para objetos relacionadas a un rostro (ojos, nariz, boca, etc.) a partir de la información previa. Capa 3 vectores de características que describen un rostro completo y capa de salida para la clasificación acorde a la tarea requerida. **Fuente:** https://towardsdatascience.com/a-comprehensive-hands-on-guide-to-transfer-learning-with-real-world-applications-in-deep-learning-212bf3b2f27a.

Los métodos más utilizados para el aprendizaje profundo están basados en las siguientes cuatro categorías principales [31], [33]:

- **Redes neuronales convolucionales (CNNs):** AlexNet [34], GoogleNet [35] entre otras.
- **Maquinas restringidas de Boltzmann (RBMs):** Redes de Creencia profunda, Máquinas de Boltzmann profundas, Modelos de energía profundos [33].
- **Codificadores automáticos (*Auto encoders*):** dispersos, sin ruido y contractivos [36].
- **Codificador disperso (*Sparse coding*):** Laplaciano, Super-Vector, entre otros [37].

El aprendizaje profundo tiene importantes aplicaciones dentro del procesamiento de lenguaje natural [38], visión por computadora [34], diagnóstico médico [39], entre varios más.

5.4 Aprendizaje por refuerzo

El aprendizaje por refuerzo (RL) se enfoca en *qué hacer* (analizar situaciones y tomar acciones) buscando maximizar una señal numérica de *recompensa*. En este método de aprendizaje, el *aprendiz* (algoritmo) desconoce las acciones que debe tomar, por tanto, va descubriendo aquellas que le permiten obtener mayor recompensa. El RL se fundamenta en la idea de aprendizaje mediante la interacción directa con un entorno, la cual produce información de tipo *causa-efecto* sobre las consecuencias de las acciones y sobre qué hacer para lograr el objetivo planteado [30].

A lo largo de la vida, un ser humano está expuesto a distintas experiencias que pueden aportar mayor claridad con relación a este tipo de aprendizaje: cuando aprende a escribir o dibujar, cuando aprende a conducir un auto, cuando aprende a manipular un nuevo dispositivo electrónico e incluso en las interacciones sociales. En todos estos casos, se requiere atención a cómo responde el entorno en función de lo que se hace, es decir, constantemente se busca influir en lo que sucede mediante el comportamiento. El RL está formalizado por la teoría de sistemas dinámicos [40] relacionado al control óptimo mediante los procesos de decisión de Markov inciertos [41]. Por tanto, es un problema de decisión dentro de los denominados *problemas de decisión secuencial bajo incertidumbre* [42].

Al RL se lo considera *secuencial*, ya que, el problema está formado por varios subproblemas en secuencia, donde el agente elige una acción en cada etapa de la secuencia, considerando tanto la consecuencia actual como las futuras para la resolución del problema. Y está bajo *incertidumbre*, ya que las consecuencias de las acciones no están disponibles previamente o de manera determinista, sino que el conocimiento lo va formando durante las interacciones (experiencia), ya que el agente es capaz de sensar el estado del ambiente y tomar las medidas adecuadas para afectarlo (**Fig. 5.6**) [42].

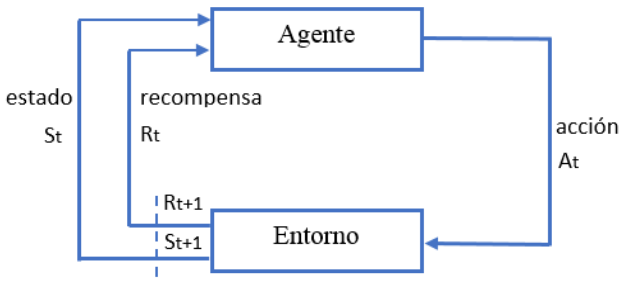


Fig. 5.6: Esquema del proceso de aprendizaje de un agente mediante la interacción directa con el entorno. El agente afecta al entorno mediante una acción, el cual, responde mediante el cambio de estado y una señal de recompensa que indica que tan buena fue la acción tomada.

Una de las consideraciones más importantes para tener en cuenta en el aprendizaje por refuerzo, es el balance entre *exploración y explotación.* Como el sistema depende de la elección de acciones que permiten obtener la mayor recompensa posible, éste debe preferir aquellas que ya ha probado y han sido exitosas. En la explotación el agente elige las acciones ya probadas previamente y que han permitido obtener una buena recompensa, mientras que, en la exploración el agente elige nuevas acciones con la finalidad de buscar mejores resultados. Durante un proceso estocástico, cada acción se ejecuta varias veces para obtener una estimación confiable de su recompensa esperada [42].

5.4.1 Elementos del aprendizaje por refuerzo

En la **Fig. 5.6** se muestran los elementos que componen un sistema inteligente entrenado mediante RL**.** De acuerdo con Sutton et al. en [30] los elementos son: el *agente*, el *entorno*, el *estado*, la *política*, la *señal de recompensa*, la *función valor* y en ciertos casos el *modelo* del entorno.

- **Agente:** entidad física o virtual formada por un conjunto de elementos, reglas y procedimientos que le permiten interactuar en un determinado entorno.
- **Entorno:** ambiente físico o virtual en el que un agente opera.
- **Estado:** situación o etapa actual en la que se encuentra el agente en relación con el problema de aprendizaje que está abordando.
- **Política:** generalmente son estocásticas y definen el comportamiento del agente de acuerdo con sus condiciones actuales dentro del entorno; es decir, que mapea los estados percibidos en acciones. En psicología esto se conoce como el conjunto de reglas o asociaciones de estímulo-respuesta. La política es el núcleo del agente durante su aprendizaje, ya que, ésta es suficiente para definir su comportamiento. Una política puede ser una función simple, una tabla de búsqueda o un proceso de búsqueda.
- **Señal de recompensa:** define el objetivo a alcanzar dentro del problema de aprendizaje. En cada interacción del agente, el entorno le envía esta señal numérica donde el agente busca obtener el máximo valor total a lo largo de la trayectoria ejecutada. Está relacionada con las señales de dolor y placer que se producen en los sistemas biológicos. La señal de recompensa es la que permite alterar la política, en el caso de que una acción seleccionada por la política le sigue una recompensa baja, entonces, se cambia para seleccionar otra acción en esa situación en el futuro. Esta señal indica qué tan buenas son las acciones tomadas en el sentido inmediato.
- **Función valor:** es la cantidad total de recompensa que un agente puede esperar acumular en el futuro, a partir de un estado; es decir, permite evaluar la recompensa a largo plazo. Puede ser el caso donde un estado puede producir una recompensa inmediata baja, pero tener un valor alto porque está seguido regularmente por otros estados que producen recompensas altas o viceversa. Por tanto, se buscan las acciones que permitan obtener la más alta *función valor*, puesto que esto permitirá obtener la mayor recompensa.
- **Modelo:** permite imitar el comportamiento del entorno para hacer inferencias sobre el comportamiento del entorno. Por ejemplo, dado un estado y una acción, el modelo podría predecir el siguiente estado resultante y la próxima recompensa. Cuando los problemas permiten modelar su entorno se los conoce como *basados en un modelo*, en cuyo caso, el agente decide su comportamiento considerando posibles situaciones posibles. Las aplicaciones están dadas en los sistemas de planeación. Cuando no se cuenta con un modelo se lo conoce como *libre de modelo*, en cuyo caso, el agente aprende directamente mediante prueba-error.

5.4.2 Los procesos de decisión de Markov

El aprendizaje por refuerzo usa como marco de referencia formal los procesos de decisión de Markov (MDPs) para definir la interacción entre el agente y su entorno en términos de acciones, estados y recompensas. Los MDPs son una formalización clásica para la toma de decisiones secuenciales, donde, las acciones influyen tanto en las recompensas inmediatas como en los estados subsiguientes y, por ende, en las recompensas futuras [30].

De esta manera, el aprendizaje por refuerzo se distingue de otros métodos como los algoritmos evolutivos, de enjambres y algoritmos de simulación. Aunque estos últimos también pueden crear políticas óptimas, no utilizan una estructura temporal que le permita al agente conocer los estados por los que pasa y las acciones que selecciona [30], [42].

Un MDP finito está conformado por la tupla $\langle S, A, T, R, E\rangle$, donde S es el espacio de estados del agente, A el espacio de acciones, T la dinámica de las transiciones $T(s, a, s')$ que retorna la probabilidad de seleccionar una acción a, en el estado actual s y obtener el siguiente estado s'; $R(s, a, s')$ es la función recompensa que se obtiene al pasar del estado s al s' seleccionado la acción a; E, donde $E \subset S$, es el conjunto de los estados terminales que evitan futuras transiciones de estado y recompensas [43].

Una propiedad importante de los MDPs es la conocida como propiedad Markov [44], que asume que las transiciones de estado son dependientes del último estado del sistema y son independientes de cualquier estado de entorno anterior o acción del agente. Esta propiedad se muestra en la **Ec. 5.1**:

$$p(s', r \mid s, a) = \Pr\{S_t = s', R_t = r \mid S_{t-1} = s, A_{t-1} = a\} \tag{5.1}$$

donde p: $S \times R \times S \times A \to [0, 1]$ es una función determinística de 4 parámetros que determina la distribución de probabilidad para cada elección de s y a. Como toda distribución de probabilidad, debe cumplir la propiedad donde la sumatoria total debe ser igual a 1:

$$\sum_{s' \in S} \sum_{r \in \mathrm{R}} p(s', r \mid s, a) = 1\,, \forall\, s \in S, a \in A(s) \tag{5.2}$$

En la **Fig. 5.7**, se muestra un esquema de los MDPs, el cual, resume la transición entre estados por medio de una acción dada una función de probabilidad junto con la señal de recompensa obtenida.

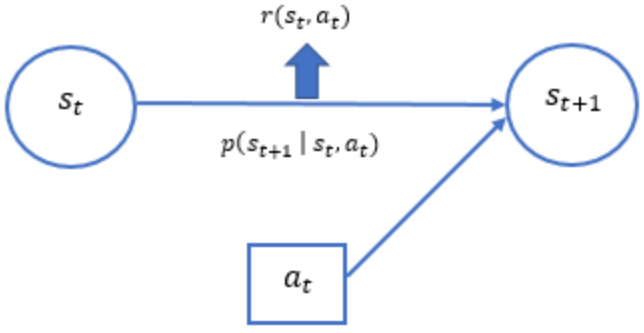


Fig. 5.7: Esquema de transición de estados basados en los procesos de decisión de Markov.

5.4.3 Retornos y episodios

Como se ha mencionado, el agente tiene como objetivo maximizar la señal de recompensa a lo largo del tiempo. Este proceso puede expresarse mediante la suma de subsecuencias de la señal de retorno [30]:

$$G_t = R_{t+1} + R_{t+2} + R_{t+3} + \cdots + R_T \tag{5.3}$$

donde G_t es el valor de retorno esperado, T es el valor de la etapa de tiempo final expresado como $T = t + n$. Este enfoque se utiliza cuando el agente puede dividir naturalmente el tiempo total de procesamiento en subsecuencias denominadas *episodios.*

El concepto adicional que es introducido a los episodios es el de *descuento* definido en la **Ec. 5.4**. Siguiendo este enfoque, el agente selecciona acciones para maximizar la suma de las recompensas descontadas a recibir en el futuro.

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots + \gamma^{n-1} R_T \tag{5.4}$$

donde γ es el parámetro conocido como *tasa de descuento* y está en el rango $0 \leq \gamma \leq 1$. Si $\gamma = 1$ la suma total está basada en los episodios en los que se encuentra dividido el problema de aprendizaje. Si $\gamma = 0$ el sistema solo está basado en las recompensas inmediatas y no puede prever los valores de recompensas futuros. Por tanto, γ evalúa cuán valiosa es la recompensa inmediata en términos de la recompensa final buscada [45].

5.4.4 Políticas y Función valor

La función valor o la función de un estado permite al agente estimar *cuán bueno* es realizar una acción en un estado dado. La expresión "cuán bueno" se define con base en los valores de las futuras recompensas, es decir, el retorno esperado. La función valor depende de la forma particular que tiene el agente para actuar, lo cual depende de la *política* [30].

De manera formal, una política π mapea los estados a probabilidades de selección de cada acción posible. Los métodos de aprendizaje por refuerzo muestran cómo la política va cambiando de acuerdo con la experiencia. El valor de retorno esperado de un estado s bajo una política π se denota como $v_\pi(s)$ y está definido según:

$$v_\pi(s) = \mathbb{E}_\pi(G_t | S_t = s) = \mathbb{E}_\pi\left(\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s\right), \forall\, s \in S \tag{5.5}$$

donde $\mathbb{E}_\pi(.)$ es el valor de retorno esperado y v_π se conoce también como la función *estado-valor* siguiendo una política π.

Sin embargo, cuando el modelo de transiciones es desconocido, el cálculo de la función v no permite encontrar la política π. Para esto, Watkins en [46] propone calcular el valor de retorno esperado para una acción a en un estado s bajo una política π denotada como $q_\pi(s, a)$:

$$q_\pi(s, a) = \mathbb{E}_\pi(G_t | S_t = s, A_t = a) = \mathbb{E}_\pi\left(\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s, A_t = a\right) \tag{5.6}$$

donde q_π se conoce como la función *acción-valor* siguiendo una política π.

Los algoritmos RL estiman estas funciones mediante la programación dinámica y una aproximación por los métodos Monte-Carlo. La programación dinámica [47] permite al agente calcular las políticas óptimas cuando este conoce el modelo de todas las transiciones y los valores

de recompensa (MDPs). Los métodos Monte-Carlo [48], al ser no locales, permiten al agente el cálculo de los parámetros óptimos de aprendizaje sin conocer previamente el modelo del ambiente dado por los MDPs [42].

En general, los algoritmos de RL se pueden dividir en dos categorías [43]:

- **En línea:** basados en interacción donde la actualización de los valores de la función estado-valor y acción valor se realizan durante las transiciones de estado a estado. Algoritmos de esta categoría son: Q-learning [46], SARSA [30] o Política de gradiente [49].
- **Sin conexión:** la actualización de los valores de las funciones no se realiza durante las transiciones. Algoritmos de esta categoría son: Iteración de Políticas por mínimos cuadrados [50] o Q-iteraciones ajustadas [51]

Un detalle más completo de los algoritmos RL ha sido presentado por C. Szepesvári [52] y L. Busoniu et al. [53]. A continuación, se describen los algoritmos de diferencia temporal (TD) que son la base para las contribuciones realizadas por este trabajo.

5.4.5 Métodos de diferencia temporal (TD)

Los métodos de diferencia temporal se consideran el corazón del aprendizaje por refuerzo, puesto que, han dado lugar a la mayoría de los algoritmos utilizados por este paradigma. Los métodos TD combinan las propiedades de la programación dinámica y de los métodos Monte-Carlo. Mientras que la programación dinámica requiere conocer exactamente el modelo MDPs de transiciones del ambiente, lo cual puede resultar, en muchos casos, costoso a nivel computacional, los métodos Monte-Carlo requieren solo la *experiencia* de las interacciones entre el agente y su entorno [30].

Los métodos Monte-Carlo [46] consisten en realizar el mayor número de trayectorias entre todos los estados $s \in S$, mientras se estima la función estado-valor $v_\pi(s)$, siguiendo una política π, mediante el promedio de las recompensas acumuladas observadas a lo largo de dichas trayectorias. En cada prueba, el agente registra las transiciones realizadas y las recompensas obtenidas para actualizar los valores de la función estado-valor de acuerdo con el esquema de recompensa con descuento (Sección 5.4.3), ya que estos métodos son solo aplicables cuando el problema se puede separar en episodios [42]. Usando estos métodos, el cálculo de la función $v_\pi(s)$ se escribe como:

$$v_\pi(s_t) = v_\pi(s_t) + \alpha(s_t)\big(R_{t+1} + R_{t+2} + \cdots + R_N - v_\pi(s_t)\big) \tag{5.7}$$

donde $\alpha(s_t)$ es la tasa de aprendizaje que converge a cero a lo largo de las iteraciones, N es el número total de estados que conforman una trayectoria y la diferencia estima el error obtenido durante las actualizaciones, ya que en cada iteración el valor de la función $v_\pi(s)$ es estimado cuando se sigue la política π.

La **Ec. 5.7** muestra el cálculo de la función estado-valor cuando el agente termina la trayectoria; sin embargo, los métodos de diferencia temporal han mejorado este procedimiento, permitiendo calcular esta función después de cada transición. Por tanto, la regla de actualización puede ser expresada como:

$$v_\pi(s_t) = v_\pi(s_t) + \alpha(s_t)\,(\delta_t + \delta_{t+1} + \delta_{t+2} + \cdots + \delta_{t+N}) \tag{5.8}$$

donde δ_t es el error temporal, que se calcula en cada estado por $\delta_t = R_{t+1} + v_\pi(s_{t+1}) - v_\pi(s_t)$ para $t = 0, 1, \dots, N-1$, el cual, es interpretado como la diferencia entre el valor actual $v_\pi(s_t)$ y la estimación corregida $R_{t+1} + v_\pi(s_{t+1})$.

Este cálculo se realiza una vez que termina una transición de estado (s, r, s'), donde mediante esta regla de actualización de los valores $v_\pi(s_t)$ se da lugar a la categoría de algoritmos conocida como "en línea", en los cuales la actualización combina las propiedades incrementales de la programación dinámica y la experiencia del agente mediante los métodos Monte-Carlo [42].

5.4.5.1 Algoritmo TD (0)

Es uno de los algoritmos de aprendizaje para entornos de modelo *libre*. Su aprendizaje se basa en *bootstrapping* [54] mediante la estimación de la función de estado-valor con base en la **Ec. 5.8**, la cual, es modificada para que el algoritmo sea *en línea*, considerando solo 2 estados sucesivos en el tiempo como lo muestra la **Ec. 5.9.**

$$v_\pi(s_t) = v_\pi(s_t) + \alpha(s_t)\,\delta_t \tag{5.9}$$

Dentro del parámetro δ_t, el término $R_{t+1} + v_\pi(s_{t+1})$ se conoce como objetivo TD, debido a la diferencia entre dos valores de funciones de estado-valor sucesivas en el tiempo [52]. El indicador 0 es debido a que este algoritmo no utiliza huellas de elegibilidad[13] en los estados. Esta elegibilidad permite que el agente transite durante el proceso de aprendizaje por los mejores estados descubiertos [42].

El pseudocódigo de TD (0) se resume en el **Algoritmo 5.1** presentado por Sutton et.al en [30].

Algoritmo 5.1: Pseudocódigo de TD (0)

Con la política π a ser evaluada y un valor arbitrario inicial para $V(s)$, generalmente inicializado en cero.

- **for** t **in range (**$episodios$**):**
- Seleccionar el estado s_t
- **While** s_t **is not** terminal:
- Tomar una acción a con el estado actual s_t
- Observar el estado s_{t+1} y el valor de recompensa r
- Calcular el error $\delta_t = R_{t+1} + v_\pi(s_{t+1}) - v_\pi(s_t)$
- Actualizar la función estado-valor. (**Ec. 5.9**)
- Asignar $s_t = s_{t+1}$

La convergencia del algoritmo TD (0) está basada en dos procesos acoplados: el primero estima la recompensa inmediata en cada estado y el segundo aproxima la función de valor resultante de estas estimaciones mediante la propagación a lo largo de las transiciones [42].

Una de las variaciones de este algoritmo es TD (λ) propuesto por Sutton en [55] que utiliza huellas de elegibilidad en los estados. Este algoritmo ha sido aplicado por G. Tesauro para la creación de TD-Gammon [56]. TD (λ) no es un algoritmo en línea; sin embargo, Van Seijen et al. [57] propone una versión para solucionar este aspecto.

[13] En inglés se lo conoce como *eligibility traces*.

5.4.5.2 Algoritmo SARSA

El algoritmo SARSA está basado en el cálculo de la función acción-valor Q propuesta por Watkins en [46] cuando el modelo de transiciones es desconocido. La función Q aporta un conocimiento similar al de la función v, la cual se encuentra definida en la **Ec. 5.10.**

$$Q_\pi(s,a) = r(s,a) + \gamma \sum_{s'} p(s'|s,a)\, v_\pi(s'), \qquad \forall\, s \in S, a \in A(s) \tag{5.10}$$

donde $r(s,a)$ es el valor de recompensa de la transición del estado s tomando la acción a, siendo $r \in R$ y $s' \in S^+$ siendo S^+ el mismo conjunto de estados S, incluyendo un estado terminal cuando el problema está dividido en episodios [42].

Para obtener las políticas óptimas basta con encontrar los valores óptimos de las funciones v_π y Q_π, las cuales, satisfacen las ecuaciones de optimalidad de Bellman [58]. La estimación de estos valores óptimos está basada en las siguientes ecuaciones, respectivamente en las **Ec. 5.11 y 5.12**:

$$\begin{aligned} v_*(s) &= \max_a \mathbb{E}(R_{t+1} + \gamma v_*(S_{t+1}) | S_t = s, A_t = a\,) \\ &= \max_a \sum_{s',r} p(s', r|s,a)\,[r + \gamma v_*(s')] \end{aligned} \tag{5.11}$$

$$\begin{aligned} Q_*(s) &= \mathbb{E}\left(R_{t+1} + \gamma \max_{a'} q_*(S_{t+1}, a') \,\middle|\, S_t = s, A_t = a\,\right) \\ &= \sum_{s',r} p(s', r|s,a)\left[r + \max_{a'} q_*(s', a')\right] \end{aligned} \tag{5.12}$$

donde $v_*(s)$ y $Q_*(s)$ son los valores óptimos para las funciones estado-valor y acción-valor y está dado para todo $s \in S$, $a \in A(s)$ y $s' \in S^+$.

De acuerdo con estas ecuaciones de optimalidad se puede estimar la función $v_*(s) = \max_a Q_*(s,a)$ y la política óptima se establece como $\pi_*(s) = \operatorname{argmax}_a Q_*(s,a)$. La información necesaria para actualizar los valores está dada por $(s_t,\ a_t,\ s_{t+1}, a_{t+1})$; de lo cual, proviene su nombre. El aprendizaje se basa en la función acción-valor y se calcula mediante la **Ec. 5.13**:

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha[r_{t+1} + \gamma\, Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \tag{5.13}$$

El pseudocódigo de SARSA se resume en el **Algoritmo 5.2** presentado por Sutton et.al en [30].

Algoritmo 5.2: Pseudocódigo de SARSA

Inicializar arbitrariamente $Q(s,a)$, para todo $s \in S$, $a \in A(s)$ y $Q(terminal) = 0$

- **for** t **in range (**$episodios$**):**
- Seleccionar el estado s_t
- Con el estado actual s_t, tomar una acción a usando una política π a partir
- de Q (una política muy conocida es e-greedy [30])
- **While** s_t **is not** terminal:
- Con la acción a, observar el estado s_{t+1} y el valor de recompensa r
- Seleccionar s_{t+1}, a_{t+1} siguiendo la misma política.
- Actualizar la función acción-valor. (**Ec. 5.13**)
- Asignar $s_t = s_{t+1}$ y $a_t = a_{t+1}$

Como se puede notar en el **Algoritmo 5.2**, SARSA requiere conocer previamente cuál será la acción a_{t+1} para poder hacer la actualización. Por tanto, el aprendizaje está estrechamente relacionado con la política π, lo que limita el proceso de exploración. Este algoritmo se conoce como de "política activado", lo que hace difícil probar su convergencia [42].

5.4.5.3 Algoritmo Q-learning

El algoritmo Q-learning [46] es una simplificación de SARSA. La función Q se actualiza sin considerar la acción siguiente a_{t+1}. Este cálculo se muestra en **Ec. 5.14:**

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha \left[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right] \quad (5.14)$$

El término $\max_a Q(s_{t+1}, a)$ indica que la actualización de la función Q se realiza con base en las acciones óptimas, en lugar de las acciones realizadas, lo cual simplifica el proceso.

Este algoritmo es conocido como de "política no activada". Cuando se usa políticas codiciosas o e-greedy, dado que hay que establecer un equilibrio entre la exploración y la explotación, se puede realizar una elección de acción no codiciosa sin dejar de utilizar el término máximo en la actualización [30], [42].

El pseudocódigo de Q-learning se resume en el **Algoritmo 5.3** presentado por Sutton et.al en [30].

Algoritmo 5.3: Pseudocódigo de Q-learning

Inicializar arbitrariamente $Q(s, a)$, para todo $s \in S$, $a \in A(s)$ y $Q(terminal) = 0$

- **for** t **in range (**$episodios$**):**
- Seleccionar el estado s_t
- **While** s_t **is not** terminal:
- Seleccionar a_{t+1} usando una política π a partir de Q (una política es
- e-greedy [30])
- Con la acción a_{t+1}, observar el estado s_{t+1} y el valor de recompensa r
- Actualizar la función acción-valor. (**Ec. 5.14**)
- Asignar $s_t = s_{t+1}$

Como lo muestra el **Algoritmo 5.3**, la actualización de la función Q se realiza mediante la función instantánea disponible. Esta característica hace que, este algoritmo RL, sea muy utilizado a pesar de sus limitaciones. Se han propuesto los algoritmos SARSA (λ) y Q-learning (λ) buscando mejorar el rendimiento mediante las huellas de elegibilidad en los estados [42].

Otros algoritmos complejos también están basados en Q-learning, por ejemplo, los que utilizan redes neuronales artificiales, presentados en la siguiente sección, que han dado lugar al **RL** profundo.

5.5 Aprendizaje por refuerzo profundo

El RL profundo es la combinación de los métodos de aprendizaje por refuerzo y las técnicas del aprendizaje de máquina. Este campo de investigación ha sido capaz de dar a una máquina la *habilidad* de resolver tareas complejas dentro del campo del control óptimo para la toma de decisiones secuenciales [59].

Esta habilidad se relaciona con la tarea de decidir, desde la experiencia, la secuencia de acciones adecuadas que le permitan lograr un objetivo interactuando con un ambiente desconocido. El RL profundo es muy útil en problemas que tienen un espacio de estados de alta dimensión. Su capacidad de aprender desde distintos niveles de abstracción, lo hace, muy versátil incluso cuando el conocimiento *a priori* del problema sea mínimo. Mnih en [60] propone la aplicación de un agente RL profundo para aprender con éxito a jugar video juegos Atari 2600, que tiene como entradas de percepción visual a imágenes compuestas por cientos de píxeles. Esto abrió una nueva posibilidad para la resolución de problemas, imitando el comportamiento humano, que antes era difícil de imaginar [59].

Otras aplicaciones relevantes del RL profundo en teoría de juegos se encuentran en [61], [62]. Sin embargo, las aplicaciones también llegan al campo de la robótica [63], [64], redes inteligentes [65], [66], manejo automático de automóviles [67], [68] y al diagnóstico médico [69], [70].

En la **Fig. 5.8**, se muestra el esquema que sigue el aprendizaje RL profundo, junto con sus elementos.

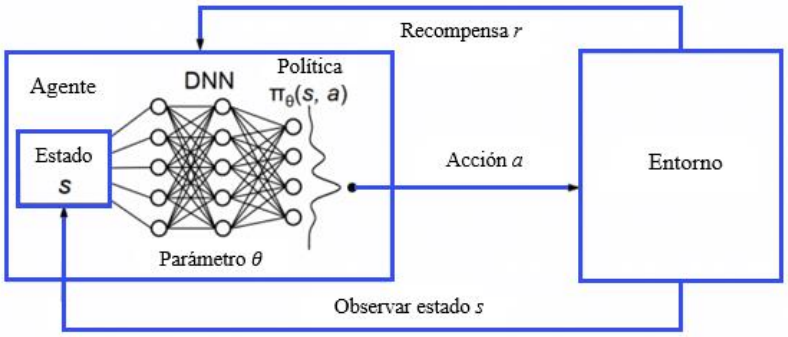


Fig. 5.8: Esquema del proceso de aprendizaje de un agente mediante aprendizaje profundo. El agente afecta al entorno mediante una acción obtenida mediante una función no lineal como una red neuronal, que tiene como entrada el valor de un estado. El entrenamiento se basa en el ajuste de los parámetros o pesos de la red neuronal. El entorno responde al agente con un valor de recompensa y el cambio de estado.

5.5.1 Algoritmo profundo Q-Network

El algoritmo profundo Q-Network (DQN) es una variación de Q-learning, en el cual, se utilizan redes neuronales multicapa como herramienta de aprendizaje de máquina. Fue propuesto por Mnih en [60], con el objetivo de entrenar un agente para jugar video juegos Atari 2600 dentro del ambiente de aprendizaje ARCADE (ALE) [71], a partir de imágenes y datos de rendimiento, utilizando la misma arquitectura de red neuronal profunda e hiperparámetros para todos los juegos.

La red neuronal tiene como entrada un estado s y obtiene como salida un vector a de valores de acciones $Q(s,.;\theta)$ donde θ son los parámetros de la red. En un espacio de estados de n dimensiones y un espacio de acciones con m posibilidades, la red neuronal es una función de $\mathbb{R}^n \rightarrow \mathbb{R}^m$. Para evitar la inestabilidad, que presenta el RL cuando se tiene como aproximador una función no lineal como las redes neuronales, Mnih hizo dos contribuciones importantes para la aproximación de la función acción-valor Q [60]:

1.- usar en el entrenamiento un procedimiento biológicamente inspirado denominado *repetición de la experiencia* (*experience replay*) [72], utilizando grupos de datos durante el entrenamiento. Esto permite evitar las correlaciones y suavizar la distribución de los datos.

2.- utilizar una red objetivo (*target network*) que actualiza periódicamente sus parámetros denominados θ^- para reducir la correlación entre los valores de la función Q obtenidos con los valores objetivos.

El agente aprende una función *acción-valor* parametrizada $Q(s, a;\ \theta_t)$, donde θ_t son los pesos de la red neuronal en la iteración t. El proceso de entrenamiento planteado requiere almacenar la experiencia del agente formado por una tupla de cinco parámetros $e_t = (s_t, a_t, r_t, s_{t+1}, T)$, donde s_t es el estado en la iteración actual t, a_t la acción, r_t la recompensa obtenida, s_{t+1} el estado siguiente y T que es un booleano que indica si corresponde al estado terminal [73].

Estas experiencias se almacenan en un conjunto de datos $D_t = \{e_1, \dots, e_t\}$, de donde se elige aleatoriamente un subgrupo denominado *mini-batch*. La función de pérdida durante el entrenamiento está basada en:

$$L_i(\theta_i) = U(D)\left[\left(r_t + \gamma \max_{a'} Q(s_{t+1}, a_{t+1}; \theta_t^-) - Q(s_t, a_t; \theta_t)\right)^2\right] \tag{5.15}$$

donde $U(D)$ es el mini-batch utilizado durante el entrenamiento en el instante t, γ es la tasa de descuento cuando el aprendizaje es dividido en episodios, θ_t^- son los parámetros de la función Q objetivo y θ_t los parámetros de la función Q en la iteración t. La actualización de estos parámetros se realiza cada C iteraciones donde $\theta^- = \theta_t$. La optimización de la red neuronal se realiza mediante el algoritmo de descenso por el gradiente [60].

El pseudocódigo del Deep Q-Network se presenta en el **Algoritmo 5.4,** el cual, es propuesto por Mnih en [60].

Algoritmo 5.4: Pseudocódigo de Deep Q-Network

Inicializar la memoria de repetición D con una capacidad N.
Inicializar la función acción-valor Q con pesos aleatorios θ
Inicializar la función acción-valor objetivo $\hat{Q}$ con pesos $\theta^- = \theta$

- **for _ in range (**$episodios$**):**
- Inicializar la secuencia $s_1 = \{x_1\}$ y la secuencia preprocesada $\emptyset_1 = \emptyset(s_1)$.
- **for** t **in range (**T**):**
- Con una probabilidad ε seleccionar una acción aleatoria a_t
- En otro caso: $a_t = argmax_a\ Q(\emptyset(s_t), a;\ \theta)$
- Con la acción a_t, observar la imagen x_{t+1} y el valor de recompensa r
- Ajustar $s_{t+1} = s_t, a_t, x_{t+1}$ y pre procesar $\emptyset_{t+1} = \emptyset(s_{t+1})$.
- Almacenar en D el conjunto de datos $(\emptyset_t, a_t, r_t, \emptyset_{t+1})$
- Seleccionar de D un mini-batch $(\emptyset_j, a_j, r_j, \emptyset_{j+1})$
- **If** episode (j+1) **is terminal**:
- $Y_j = r_j$
- **Else:**
- $Y_j = r_j + \gamma \max_{a'} \hat{Q}(\emptyset_{j+1}, a'; \theta_t^-)$
- Optimizar por descenso de gradiente $\left(Y_j - Q(\emptyset_j, a_j;\ \theta)\right)^2$ respecto de los pesos θ
- Asignar cada C iteraciones $\hat{Q} = Q$

El **Algoritmo 5.4** muestra el parámetro $\emptyset_1 = \emptyset(s_1)$ que tiene relación con las imágenes que utiliza para el entrenamiento en los video juegos. Para el preprocesamiento de las imágenes utiliza una red convolucional en la capa de entrada y el almacenamiento se realiza cada 4 nuevos *frames* para evitar una sobre-optimización [73].

5.5.2 Algoritmo profundo Doble Q-Network

Tanto en el algoritmo Q-learning como en DQN se utiliza el operador max para seleccionar y evaluar una acción. Esto casi siempre genera una sobre-optimización (*overoptimistic*) en la estimación de valores [74]. Para evitar esto, se propone separar los procesos de selección y evaluación mediante el uso de dos arquitecturas de redes neuronales acopladas.

En Doble Q-learning, propuesto por Van Hassel en [75], un agente aprende desde dos funciones distintas Q^A y Q^B mediante la asignación de distintas experiencias aleatorias lo que resulta en tener dos conjuntos de pesos distintos θ y θ'. En la actualización, un conjunto de pesos se utiliza para determinar la *política* y el otro para calcular su *valor*.

El error de la arquitectura Doble DQN se puede obtener de acuerdo con [74]:

$$Y_t^{DoubleQ} = r_{t+1} + \gamma\, Q\left(\underset{a}{\text{argmax}}\, Q(s_{t+1}, a; \theta_t);\ \theta_t'\right) \tag{5.16}$$

donde θ_t es el conjunto de pesos que permiten seleccionar una acción dentro del operador argmax y el segundo conjunto de pesos θ_t' se utilizan para evaluar el valor de su política.

El pseudocódigo del Doble DQN se presenta en el **Algoritmo 5.5,** el cual, es propuesto por Van Hassel en [75].

Algoritmo 5.5: Pseudocódigo de Doble Q-Network

Inicializar la memoria de repetición D con una capacidad N.
Inicializar la función acción-valor Q^A con pesos aleatorios θ
Inicializar la función acción-valor Q^B con pesos aleatorios θ'

- **for _ in range (**$episodios$**):**
-
- **for** t **in range (**T**):**
- Inicializar el estado s_t
- Seleccionar a_t basado en $Q^A(s,.)$ y $Q^B(s,.)$
- Observar s_{t+1} y r
- Elegir aleatoriamente la actualización de Q^A o de Q^B
- **If** update (A):
- $a^* = argmax_a\, Q^A(s_{t+1}, a)$
- $Q^A(s_t, a_t) = Q^A(s_t, a_t) + \propto \left(r + \gamma Q^B(s_{t+1},\ a^*) - Q^A(s_t, a_t)\right)$
- **Else:**
- $b^* = argmax_a\, Q^B(s_{t+1}, a)$
- $Q^B(s_t, a_t) = Q^B(s_t, a_t) + \propto \left(r + \gamma Q^A(s_{t+1},\ b^*) - Q^B(s_t, a_t)\right)$
- Asignar $s_t = s_{t+1}$

En Doble DQN, los pesos de la segunda red θ' se reemplazan por los pesos de la red objetivo θ^- periódicamente cada C iteraciones $\theta' = \theta^-$, para la evaluación de la política actual. Esto hace que el cambio de la arquitectura base DQN se mantenga, para que se pueda comparar los rendimientos y tener una sobre carga computacional mínima [74].

5.5.3 Algoritmo profundo Dueling Q-Network

Esta arquitectura de **RL** profundo fue diseñada tomando en cuenta el conjunto de estados que se pueden presentar en un problema de modelo libre (*free-model*). En este tipo de problemas, los estados y las recompensas se generan por el entorno y donde la política esta desactivada, puesto que, el agente aprende a través de su comportamiento como es el caso de la política *e-greedy* [30].

Durante el proceso de aprendizaje, se asume que siempre es prioritaria la elección de una acción, puesto que tiene influencia directa en el rendimiento del agente. Sin embargo, pueden existir casos, como los video-juegos donde en determinados estados puede ser innecesario estimar el valor de una acción. En dichos casos, la selección de una acción u otra no repercute en el rendimiento global [76].

En la arquitectura de duelo (*Dueling*) DQN, la estimación de la función *acción-valor* $Q\ (s, a;\ \theta)$ en la capa final de salida se divide en dos parámetros: el cálculo de la función *valor* y la función *ventaja*. La estimación de la función Q se realiza con:

$$Q\ (s, a;\ \theta, \alpha, \beta) = V(s; \theta, \beta) + A(s, a;\ \theta, \alpha) \tag{5.17}$$

donde la función valor $V(s)$ indica cuán bueno es estar en un determinado estado, la función ventaja $A(s, a)$ indica cuán mejor es tomar una determinada acción del conjunto total, θ son los pesos de la red neuronal completa, mientras que α, β, son los pesos que corresponden a cada función que componen la arquitectura Dueling [76].

En la **Fig. 5.9**, se muestra la representación de la arquitectura Dueling DQN y su comparación con DQN que sigue siendo la base para el entrenamiento utilizando **RL** profundo.

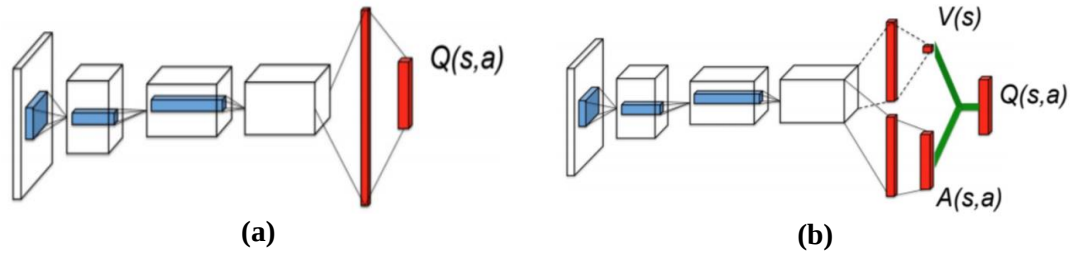


Fig. 5.9: Arquitecturas de **RL** profundo. **(a)** Arquitectura DQN [60] donde la función Q se calcula mediante un solo conjunto de pesos θ de una red neuronal multicapa. **(b)** Arquitectura Dueling DQN [76] donde la función Q se calcula mediante las funciones valor y la función ventaja considerando el conjunto de pesos total θ de la red neuronal multicapa y los pesos α, β de cada división.

La estimación de la función $V(s)$ y la función $A(s, a)$ por separado y la combinación en la capa final de salida para obtener la función Q permite al agente lograr estimaciones más sólidas respecto del *valor de estado*, ya que no está obligado a unirse a acciones específicas, lo cual, reduce significativamente la varianza [76], [77].

5.6 Contribuciones originales

Las contribuciones de este capítulo están enfocadas a la implementación de los siguientes algoritmos de RL: Q-learning, DQN, Doble DQN, Dueling DQN, los cuales, permiten que un agente aprenda las acciones y políticas óptimas que permitan registrar dos imágenes.

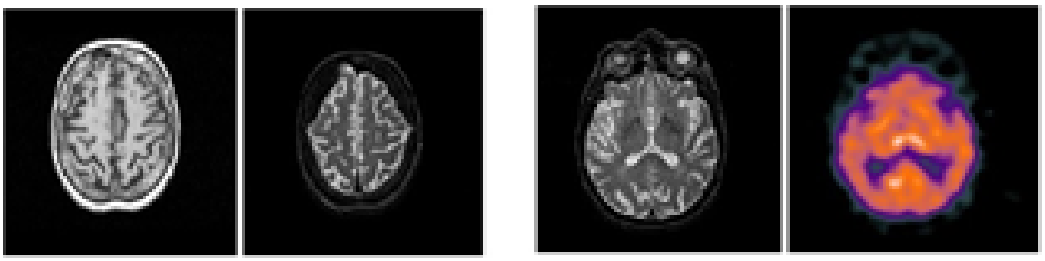
Esto ha sido publicado en Isa-Jara et al. [78]. A continuación, se detalla el desarrollo y los resultados obtenidos.

5.6.1 Entorno de registración de imágenes para algoritmos RL

El aporte de esta sección es la adaptación de los algoritmos RL antes mencionados al entorno de la registración de imágenes médicas. El tipo de registración es rígida, en modalidades monomodal y multimodal. Para esto se utilizaron imágenes de MRI de cerebro.

Descripción de las imágenes utilizadas

En la **Fig. 5.10**, se presentan las imágenes de RM en secuencias T1 y T2 con las siguientes características: 256×256 píxeles de tamaño, 8 $bits$ en escala de grises. Las imágenes SPECT tienen 3 canales definidos en el espacio RGB, 256×256 de tamaño, 8 $bits$ por cada canal del espacio color. Estas imágenes han sido provistas por el Instituto Radiológico de Mar del Plata.



(a) **(b)**

Fig. 5.10: Imágenes de MRI cerebrales utilizadas para el aprendizaje de los algoritmos **RL. (a)** Grupo 1 de imágenes pertenecientes al mismo corte en modalidades ponderadas T1 y T2. **(b)** Grupo 2 de imágenes pertenecientes al mismo corte en modalidad ponderada T2 y modalidad SPECT.

Descripción del hardware y software

Los algoritmos RL fueron implementados en *Python* v. 3.6 utilizando la librería de aprendizaje profundo *TensorFlow* v. 1.10. Las pruebas se realizaron en una computadora con sistema operativo *Debian*, con procesador Core *i5* y 8 Gb de memoria RAM.

Descripción de los elementos del RL

Los algoritmos RL han sido adaptados al entorno de registración de imágenes tomando en cuenta los elementos descritos en la *Sección 5.4.1*, bajo las siguientes consideraciones:

- **Entorno:** lo conforman las imágenes fija y móvil. Las imágenes móviles se obtienen mediante valores conocidos (**Tabla 5.1**); de esta manera es posible medir el rendimiento de cada algoritmo.
- **Estados:** el agente irá descubriendo estados definidos por el grupo de parámetros $[s, \theta, t_x, t_y]$ correspondientes a la transformación rígida, donde s es el valor de la escala, θ el valor de la rotación, t_x el valor de la traslación sobre el eje X y t_y el valor de la traslación sobre el eje Y.
- **Acciones:** el número total de acciones posibles es $2^4 = 16$, ya que existen 4 parámetros de transformación con 2 opciones posibles en cada caso (incrementar o decrementar su valor actual). El valor de paso está definido por $[\pm 0.05, \pm 0.25°, \pm 1, \pm 1]$.
- **Recompensa:** está dada por el coeficiente de correlación de Pearson para la registración monomodal y la Información Mutua en la registración multimodal. Estas medidas de similaridad serán maximizadas durante el proceso de aprendizaje. Cuando una acción produce un decremento en la medida de similaridad, el agente recibe una penalidad de -0.1. Este es el estado terminal previo, al cual, el agente reinicie el proceso.

5.6.2 Criterio de memoria de respaldo

El aporte de esta sección es la propuesta de un criterio para almacenar información en una memoria independiente denominada *memoria de respaldo*. Como se mencionó anteriormente, los algoritmos DQN se entrenan a través de lotes (mini-batchs) elegidos de manera aleatoria de una memoria denominada principal. En esta se almacenan todas las experiencias ejecutadas por el agente durante cada iteración [79]. Cuando la memoria principal se completa, las nuevas experiencias reemplazan a las almacenadas sin ninguna consideración adicional.

La *memoria de respaldo* se propone para almacenar solamente las mejores experiencias. En esta memoria, las experiencias serán reemplazadas únicamente por otras que permitan obtener mayores valores de recompensa, lo cual, evita que las mejores soluciones se pierdan a lo largo del entrenamiento.

Durante el entrenamiento se elige un mini-batch de experiencias desde la memoria principal. Para mejorar el rendimiento, se eligen posiciones aleatoriamente, tanto del mini-batch como de la memoria de respaldo, donde se reemplazará la información actual. Por lo tanto, el mini-batch contiene información de la *memoria principal* (experiencias buenas y malas) y de la *memoria de respaldo* (mejores experiencias). En este trabajo, la memoria de respaldo se usa con el algoritmo Doble DQN y los resultados muestran una mejora en el rendimiento en comparación con los otros algoritmos.

Tabla 5.1: Parámetros aplicados para generar las imágenes móviles.

Modalidad de registración	**Test 1** $[\boldsymbol{s}, \boldsymbol{\theta}, \boldsymbol{t_x}, \boldsymbol{t_y}]$	**Test 2** $[\boldsymbol{s}, \boldsymbol{\theta}, \boldsymbol{t_x}, \boldsymbol{t_y}]$
G1 T1 – T1	$[1.2, 29°, -10, 1]$	$[0.8, -29°, 10, -5]$
G1 T1 – T2	$[1.0, -15°, 0, -7]$	$[1.0, 15°, 8, 0]$
G2 T2 – T2	$[1.15, -10°, -5, -15]$	$[0.85, 10°, 7{,}12]$
G2 T2-SPECT	$[1.1, -45°, -9, 6]$	$[0.9, 45°, -17, 12]$

Detalle de las pruebas realizadas

Los parámetros iniciales se mantienen para todos los algoritmos. El número de épocas se eligió en 200 y el proceso de aprendizaje ejecuta 100 iteraciones, aunque puede terminar antes si el agente recibe una penalización. Las pruebas se evalúan a lo largo de 5 ejecuciones independientes. Las pruebas se ejecutaron de acuerdo con los parámetros mostrados en la **Tabla 5.1.**

En los resultados se muestran los promedios de los valores de la medida de similaridad por cada modalidad obtenidos durante cada iteración, junto con el valor de la recompensa acumulada. La recompensa acumulada es uno de los factores más utilizados como medida de calidad en el aprendizaje por refuerzo. Esta muestra la suma acumulada de las recompensas obtenidas durante todo el entrenamiento del agente.

Resultados obtenidos

En la **Fig. 5.11**, se muestran los resultados en función del promedio del coeficiente de correlación de Pearson obtenidos durante la registración monomodal. El rendimiento está comparado con el algoritmo doble-back_, que utiliza el criterio de memoria de respaldo propuesta.

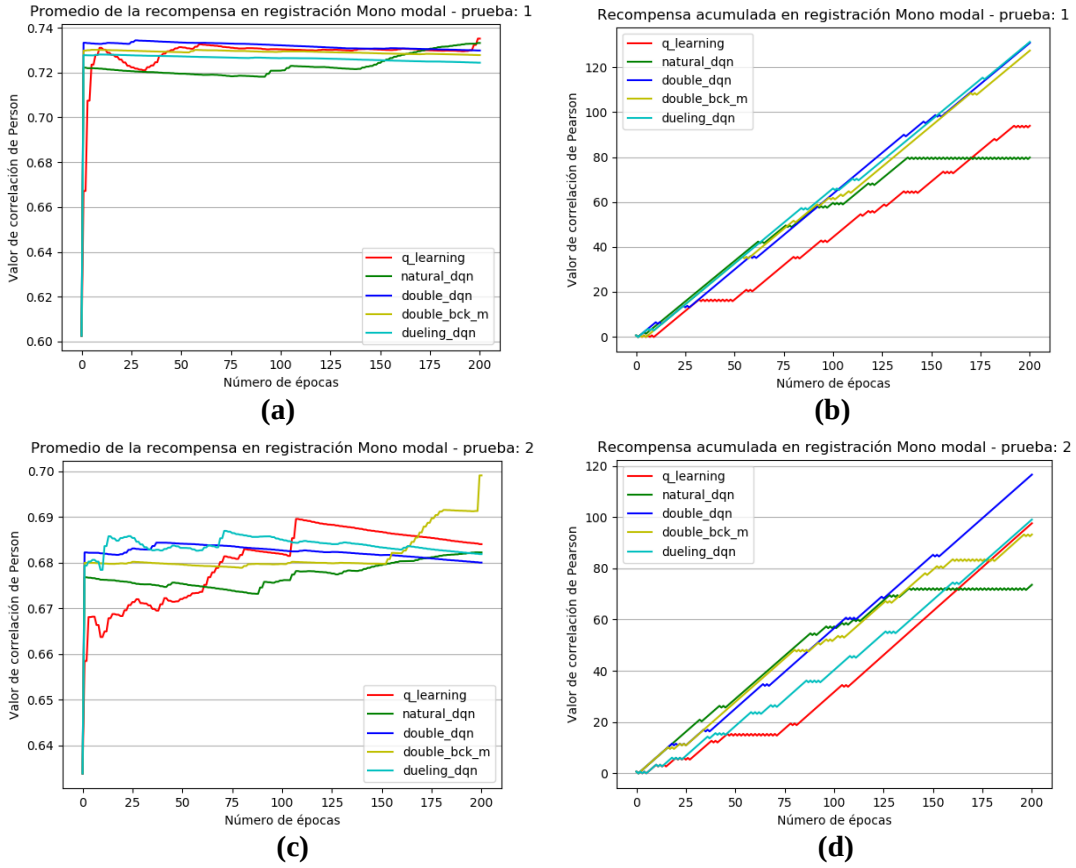


Fig. 5.11: Promedios de correlación de Pearson obtenidos durante la registración monomodal. **(a)** Valores obtenidos durante la prueba con imágenes T1-T1. **(b)** Valores de recompensa acumulada durante la prueba (a). **(c)** Valores obtenidos durante la prueba con imágenes T2-T2. **(d)** Valores de recompensa acumulada durante la prueba (c).

En la **Tabla 5.2**, se presentan los valores medios de la medida de similaridad y los errores computados durante la registración monomodal.

Tabla 5.2: Rendimiento de los algoritmos RL obtenidos en la registración monomodal.

Algoritmos RL	Prueba 1		Prueba 2	
	Corr. de Pearson	% Error	Corr. de Pearson	% Error
Q-learning	0.54	0.27	0.63	0.36
Natural-DQN	0.51	0.48	0.49	0.25
Doble-DQN	0.62	0.23	0.55	0.25
Doble-back_	0.62	0.24	0.55	0.28
Dueling-DQN	0.61	0.37	0.52	0.72

En la **Fig. 5.12**, se muestran los resultados en función del promedio de la información mutua obtenidos durante la registración multimodal.

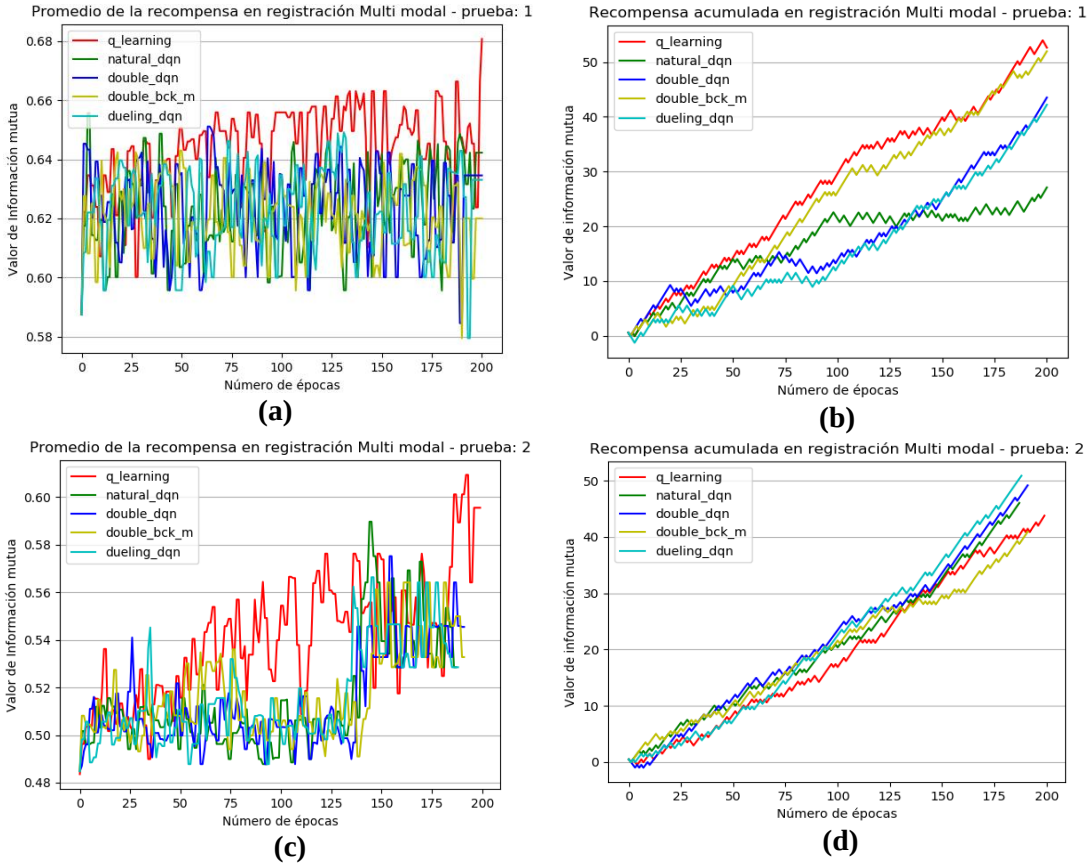


Fig. 5.12: Promedio de la información mutua obtenida durante la registración multimodal. **(a)** Valores obtenidos durante la prueba con imágenes T1-T2. **(b)** Valores de recompensa acumulada durante la prueba (a). **(c)** Valores obtenidos durante la prueba con imágenes T2-SPECT. **(d)** Valores de recompensa acumulada durante la prueba (c).

En la **Tabla 5.3**, se presentan los valores medios de la medida de similaridad y los errores computados durante la registración multimodal.

Tabla 5.3: Rendimiento de los algoritmos RL obtenidos en la registración multimodal.

Algoritmos RL	Prueba 1		Prueba 2	
	Info. mutua	% Error	Info. mutua	% Error
Q-learning	0.65	0.23	0.58	0.39
Natural-DQN	0.67	0.15	0.40	0.43
Doble-DQN	0.67	0.16	0.42	0.76
Doble-back_	0.66	0.16	0.58	0.37
Dueling-DQN	0.66	0.15	0.59	0.20

Finalmente, en la **Tabla 5.4** se muestra el promedio de tiempo utilizado durante el entrenamiento por cada algoritmo RL.

Tabla 5.4: Promedio de tiempo utilizado durante el entrenamiento.

Algoritmos	Prueba 1	Prueba 2
Q-learning	14.40 min	19.22 min
Natural-DQN	20.60 min	22.46 min
Doble-DQN	30.55 min	15.67 min
Doble-back_	15.12 min	29.71 min
Dueling-DQN	61.27 min	55.46 min

Conclusiones obtenidas a partir de los resultados

En todos los casos, los algoritmos RL dieron resultados exitosos produciendo una mejora significativa en la similitud entre las imágenes de referencia y las registradas. De acuerdo con los resultados, Q-learning, Doble-DQN y Doble-DQN con memoria de respaldo muestran mejores rendimientos respecto del resto de algoritmos analizados.

Según los resultados, el error medio obtenido por Doble-back reduce alrededor del 37% en la prueba 1 y del 35% en la prueba 2, en comparación con el error más alto alcanzado por cada algoritmo. Q-learning y Doble-DQN reducen el error medio en torno al 24% en la prueba 1 y al 19% en la prueba 2.

Además, de acuerdo con los valores de recompensa acumulada, todos los algoritmos logran mejorar los valores de correlación e información mutua durante el entrenamiento. Sin embargo, el algoritmo Doble DQN con memoria de respaldo, logra mejorar el rendimiento de los algoritmos, específicamente, durante la prueba 2 de registración monomodal como lo muestra las **Fig. 5.10 (b) y (c)** y la prueba 1 de registración multimodal en las **Fig. 5.11 (a) y (b),** lo cual permite tener una buena perspectiva de este aporte.

Debido a la complejidad de estos algoritmos, el tiempo es un factor importante para tener en cuenta. Este análisis está relacionado tanto para tiempo utilizado durante el proceso de registración, como para el tiempo de uso de los recursos computacionales disponibles. De acuerdo con los valores presentados en la **Tabla 5.4**, el tiempo en promedio es razonable para este proceso; sin embargo, no debe dejar de considerarse.

Referencias

[1] S. Russell and P. Norving, *Artificial Intelligence: A Modern Approach*, 3rd ed. Prentice Hall, 2010.

[2] A. K. Kordon, *Applying computational intelligence : how to create value*. Berlin; London: Springer, 2010.

[3] D. Poole, A. Mackworth, and R. Goebel, "Computational Intelligence and Knowledge," in *Computational Intelligence: A Logical Approach*, New York: Oxford University Press, 1998, pp. 1–22.

[4] A. P. Engelbrecht, "Computational intelligence: An introduction," in *Studies in Computational Intelligence*, 2nd ed., 2007.

[5] H. M. Said and A.-B. M. Salem, "Exploiting Computational Intelligence Paradigms in e-Technologies and Activities," *Procedia Comput. Sci.*, vol. 65, pp. 396–405, 2015.

[6] M. Mishra and M. Srivastava, "A view of Artificial Neural Network," in *2014 International Conference on Advances in Engineering & Technology Research (ICAETR - 2014)*, 2014, pp. 1–3.

[7] R. Livni, S. Shalev-Shwartz, and O. Shamir, "On the Computational Efficiency of Training Neural Networks," *Adv. neural Inf. Process. Syst.* , vol. 1, 2014.

[8] C. Freksa, "Fuzzy Systems in AI: An Overview," in *Fuzzy-Systems in Computer Science*, Wiesbaden: Vieweg+Teubner Verlag, 1994, pp. 155–169.

[9] J. Alcala-Fdez and J. M. Alonso, "A Survey of Fuzzy Systems Software: Taxonomy, Current Research Trends, and Prospects," *IEEE Trans. Fuzzy Syst.*, vol. 24, no. 1, pp. 40–56, Feb. 2016.

[10] P. . Fleming and R. . Purshouse, "Evolutionary algorithms in control systems engineering: a survey," *Control Eng. Pract.*, vol. 10, no. 11, pp. 1223–1241, Nov. 2002.

[11] M. H. Yar, V. Rahmati, and H. R. D. Oskouei, "A Survey on Evolutionary Computation: Methods and Their Applications in Engineering," *Mod. Appl. Sci.*, vol. 10, no. 11, p. 131, Aug. 2016.

[12] F. Yang, P. Wang, Y. Zhang, L. Zheng, and J. Lu, "Survey of swarm intelligence optimization algorithms," in *2017 IEEE International Conference on Unmanned Systems (ICUS)*, 2017, pp. 544–549.

[13] R. Soto, E. Rodriguez-Tello, and E. Monfroy, "Recent Advances on Swarm Intelligence for Solving Complex Engineering Problems," *Math. Probl. Eng.*, vol. 2018, pp. 1–1, Dec. 2018.

[14] V. N. Vapnik, "An Overview of Statistical Learning Theory," *IEEE Trans. NEURAL NETWORKS*, vol. 10, no. 5, pp. 988–999, 1999.

[15] R. Gholami and N. Fakhari, "Support Vector Machine: Principles, Parameters, and Applications," in *Handbook of Neural Computation*, Academic Press, 2017, pp. 515–535.

[16] A. Bharadwaj and S. Minz, "Hybrid Approach for Classification using Support Vector Machine and Decision Tree," *Proc. Intl. Conf. Adv. Electron. Electr. Comput. Sci. Eng. — EEC* , pp. 337–341, 2012.

[17] K. V. Shihabudheen and G. N. Pillai, "Recent advances in neuro-fuzzy system: A survey," *Knowledge-Based Syst.*, vol. 152, pp. 136–162, Jul. 2018.

[18] K. Dimililer and A. İlhan, "Effect of Image Enhancement on MRI Brain Images with Neural Networks," *Procedia Comput. Sci.*, vol. 102, pp. 39–44, Jan. 2016.

[19] V. Boskovitz and H. Guterman, "An adaptive neuro-fuzzy system for automatic image segmentation and edge detection," *IEEE Trans. Fuzzy Syst.*, vol. 10, no. 2, pp. 247–262, Apr. 2002.

[20] B. Xue, M. Zhang, W. N. Browne, and X. Yao, "A Survey on Evolutionary Computation Approaches to Feature Selection," *IEEE Trans. Evol. Comput.*, vol. 20, no. 4, pp. 606–626, Aug. 2016.

[21] M. K. S. Varma, N. K. K. Rao, K. K. Raju, and G. P. S. Varma, "Pixel-Based Classification Using Support Vector Machine Classifier," in *2016 IEEE 6th International Conference on Advanced Computing (IACC)*, 2016, pp. 51–55.

[22] P. Wittek, "Machine Learning," in *Quantum Machine Learning*, Academic Press, 2014, pp. 11–24.

[23] V. N. Vapnik, "Introduction: Four Periods in the Research of the Learning Problem," in *The Nature of Statistical Learning Theory*, New York, NY: Springer New York, 2000, pp. 1–15.

[24] D. Xu and Y. Tian, "A Comprehensive Survey of Clustering Algorithms," *Ann. Data Sci.*, vol. 2, no. 2, pp. 165–193, Jun. 2015.

[25] K. M. A. Patel and P. Thakral, "The best clustering algorithms in data mining," in *2016 International Conference on Communication and Signal Processing (ICCSP)*, 2016, pp. 2042–2046.

[26] M. R. M. Talabis *et al.*, "Analytics Defined," in *Information Security Analytics*, Syngress, 2015, pp. 1–12.

[27] D. Jiang, "A SOM algorithm based procedure for MRI image processing under significant Rician noise," in *2013 Australian Control Conference*, 2013, pp. 14–19.

[28] K. Arai, "Image Clustering Method Based on Self Organization Mapping: SOM Derived Density Maps and Its Application for Landsat Thematic Mapper Image Clustering," *IJARAI) Int. J. Adv. Res. Artif. Intell.*, vol. 2, no. 5, 2013.

[29] V. Chaudhary, R. S. Bhatia, and A. K. Ahlawat, "A novel Self-Organizing Map (SOM) learning algorithm with nearest and farthest neurons," *Alexandria Eng. J.*, vol. 53, no. 4, pp. 827–831, Dec. 2014.

[30] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, Second. 2017.

[31] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, and M. S. Lew, "Deep learning for visual understanding: A review," *Neurocomputing*, vol. 187, pp. 27–48, Apr. 2016.

[32] I. Goodfellow, Y. Bengio, and A. Courville, "Introduction," in *Deep Learning*, MIT Press, 2016, pp. 1–26.

[33] H.-I. Suk, "An Introduction to Neural Networks and Deep Learning," in *Deep Learning for Medical Image Analysis*, Elsevier, 2017, pp. 3–24.

[34] A. Krizhevsky, A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Adv. Neural Inf. Process. Syst.*, p. 2012.

[35] C. Szegedy *et al.*, "Going deeper with convolutions," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1–9.

[36] I. Goodfellow, Y. Bengio, and A. Courville, "Autoencoders," in *Deep Learning*, MIT Press, 2016, pp. 499–523.

[37] H. Cheng, "Robust Sparse Representation, Modeling and Learning," 2015, pp. 91–115.

[38] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," *Adv. Neural Inf. Process. Syst.*, Oct. 2013.

[39] R. R. Agravat and M. S. Raval, "Deep Learning for Automated Brain Tumor Segmentation in MRI Images," in *Soft Computing Based Medical Image Analysis*, Academic Press, 2018, pp. 183–201.

[40] N. Gilbert, D. Anzola, P. Johnson, C. Elsenbroich, T. Balke, and O. Dilaver, "Self-Organizing Dynamical Systems," in *International Encyclopedia of the Social & Behavioral Sciences*, 2nd., Elsevier, 2015, pp. 529–534.

[41] W. T. Scherer, S. Adams, and P. A. Beling, "On the Practical Art of State Definitions for Markov Decision Process Construction," *IEEE Access*, vol. 6, pp. 21115–21128, 2018.

[42] O. Sigaud and O. Buffet, *Markov Decision Processes in Artificial Intelligence*. 2010.

[43] E. Mocanu, P. H. Nguyen, and M. Gibescu, "Deep Learning for Power System Data Analysis," in *Big Data Application in Power Systems*, Elsevier, 2018, pp. 125–158.

[44] A. A. Markov, "The theory of algorithms," *Colect. Artic. To Sixtieth Birthd. Acad. Ivan Matveevich Vinograd.*, vol. 38, pp. 176–189, 1951.

[45] R. Bellman, "A markovian decision process.," 1957.

[46] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, no. 3–4, pp. 279–292, May 1992.

[47] A. G. Barto, "Reinforcement Learning and Dynamic Programming," *IFAC Proc. Vol.*, vol. 28, no. 15, pp. 407–412, Jun. 1995.

[48] S. Theodoridis, "Monte Carlo Methods," in *Machine Learning. A Bayesian and Optimization Perspective*, Academic Press, 2015, pp. 707–744.

[49] R. S. Sutton, D. Mcallester, S. Singh, and Y. Mansour, "Policy Gradient Methods for Reinforcement Learning with Function Approximation," *Adv. Neural Inf. Process. Syst.*, vol. 12, pp. 1057–1063, 2000.

[50] M. Lagoudakis and R. Parr, "Model-Free Least-Squares Policy Iteration," *Adv. neural Inf. Process. Syst.* , 2001.

[51] M. Riedmiller, "Neural Fitted Q Iteration – First Experiences with a Data Efficient Neural Reinforcement Learning Method," Springer, Berlin, Heidelberg, 2005, pp. 317–328.

[52] C. Szepesvári, "Algorithms for Reinforcement Learning," *Synth. Lect. Artif. Intell. Mach. Learn.*, vol. 4, no. 1, pp. 1–103, Jan. 2010.

[53] L. Busoniu, D. Ernst, B. De Schutter, and R. Babuska, "Approximate reinforcement learning: An overview," in *2011 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, 2011, pp. 1–8.

[54] B. Efron, "Second Thoughts on the Bootstrap," *Stat. Sci.*, vol. 18, no. 2, pp. 135–140, May 2003.

[55] R. S. Sutton, "Learning to predict by the methods of temporal differences," *Mach. Learn.*, vol. 3, no. 1, pp. 9–44, Aug. 1988.

[56] G. Tesauro, "Temporal difference learning and TD-Gammon," *Commun. ACM*, vol. 38, no. 3, pp. 58–68, Mar. 1995.

[57] H. Van Seijen and R. Sutton, "True Online TD(λ)," in *ICML'14* , 2014.

[58] R. Bellman, "On the Theory of Dynamic Programming," *Proc. Natl. Acad. Sci.*, vol. 38, no. 8, pp. 716–719, Aug. 1952.

[59] V. François-Lavet *et al.*, "An Introduction to Deep Reinforcement Learning," *Found. Trends Mach. Learn.*, vol. 11, no. 3–4, 2018.

[60] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.

[61] D. Silver *et al.*, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, Jan. 2016.

[62] N. Brown, T. Sandholm, and B. Amos, "Depth-limited solving for imperfect-information games," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 2018, pp. 7674–7685.

[63] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-End Training of Deep Visuomotor Policies," *J. Mach. Learn. Res.*, vol. 17, no. 39, pp. 1–40, Apr. 2016.

[64] L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba, and P. Abbeel, "Asymmetric Actor Critic for Image-Based Robot Learning," in *Robotics: Science and Systems XIV*, 2018.

[65] E. Mocanu, P. H. Nguyen, and M. Gibescu, "Deep Learning for Power System Data Analysis," in *Big Data Application in Power Systems*, Elsevier, 2018, pp. 125–158.

[66] L. Xiao, X. Xiao, C. Dai, M. Pengy, L. Wang, and H. V. Poor, "Reinforcement Learning-based Energy Trading for Microgrids," Jan. 2018.

[67] S. Wang, D. Jia, and X. Weng, "Deep Reinforcement Learning for Autonomous Driving," Nov. 2018.

[68] X. Pan, Y. You, Z. Wang, and C. Lu, "Virtual to Real Reinforcement Learning for Autonomous Driving," Apr. 2017.

[69] N. Liu, Y. Liu, B. Logan, Z. Xu, J. Tang, and Y. Wang, "Deep Reinforcement Learning for Dynamic Treatment Regimes on Medical Registry Data," Jan. 2018.

[70] A. Jonsson, "Deep Reinforcement Learning in Medicine," *Kidney Dis.*, vol. 5, no. 1, pp. 18–22, Feb. 2019.

[71] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling, "The Arcade Learning Environment: An Evaluation Platform for General Agents," *J. Artif. Intell. Res.*, vol. 47, pp. 253–279, 2012.

[72] L.-J. Lin, "Self-improving reactive agents based on reinforcement learning, planning and teaching," *Mach. Learn.*, vol. 8, no. 3–4, pp. 293–321, May 1992.

[73] M. Roderick, J. MacGlashan, and S. Tellex, "Implementing the Deep Q-Network," Nov. 2017.

[74] H. van Hasselt, A. Guez, and D. Silver, "Deep Reinforcement Learning with Double Q-learning," Sep. 2015.

[75] H. Van Hasselt, "Double Q-learning," *Adv. Neural Inf. Process. Syst. 23 NIPS*, 2010.

[76] Z. Wang, T. Schaul, M. Hessel, M. Com, H. Van Hasselt, and M. Lanctot, "Dueling Network Architectures for Deep Reinforcement Learning."

[77] M. E. Harmon, L. C. Baird, and A. H. Klopr, "Advantage Updating Applied to a Differential Game," *NIPS*, 1995.

[78] R. F. Isa-Jara, G. J. Meschino, and V. L. Ballarin, "A Comparative Study of Reinforcement Learning Algorithms Applied to Medical Image Registration," VIII Latin American Conference on Biomedical Engineering and XLII National Conference on Biomedical Engineering, Springer International Publishing, 2020, pp. 281–289.

[79] E. Hoffer, I. Hubara, and D. Soudry, "Train longer, generalize better: closing the generalization gap in large batch training of neural networks."

Capítulo 6 Conclusiones

> Aprende de cuanto te pueden enseñar todas las personas que caminan a tu lado. Con esfuerzo, constancia e ilusión, TODO en esta vida se puede lograr, pero hay que desearlo con todas las fuerzas.

En esta Tesis se abordó el problema de la registración de imágenes, específicamente, para el caso de imágenes médicas utilizando técnicas de procesamiento de imágenes y algoritmos de inteligencia computacional, desde la perspectiva de la optimización estocástica y del aprendizaje de máquina. Estos enfoques han sido la base para las contribuciones realizadas, obteniendo resultados experimentales prometedores, los cuales, pueden servir como base para futuras investigaciones en este campo.

6.1 Conclusiones específicas

A continuación, se detallan las conclusiones específicas obtenidas de acuerdo a los enfoques utilizados en cada capítulo.

Capítulo 3 - Optimización

Los algoritmos de inteligencia de enjambres presentan un buen rendimiento al ser aplicados en registración de imágenes tanto en monomodo como en multimodo, utilizando directamente la información de sus píxeles.

Mediante la propuesta de una metaheurística enfocada en registración se logra mejorar el rendimiento del algoritmo PSO. El banco de pruebas realizado utiliza distinta cantidad de partículas en las poblaciones iniciales del algoritmo original y propuesto. Acorde a los resultados, en el caso del PSO propuesto las poblaciones a partir de 30 partículas no tienen impacto significativo en el rendimiento, puesto que se logra obtener un promedio en la medida de similaridad estable en el rango $[0.65 - 0.70]$. Esto lo hace muy versátil y óptimo para el uso eficiente de los recursos computacionales.

Con la contribución de un nuevo algoritmo de inteligencia colectiva (LiA) se propone una prometedora y novedosa solución a los problemas de optimización global. El rendimiento de LiA también ha sido probado con la librería estándar COCO (BBOB 2019), lo cual permite compararlo con otros algoritmos aceptados en este campo científico.

Los resultados obtenidos por LiA en registración de imágenes, y al compararlos con los algoritmos PSO, ACO, FWA y ABC, permiten concluir que tiene un rendimiento apto para este tipo de aplicaciones e incluso los mejora notablemente, en algunos casos.

Capítulo 4 - Extracción de características

El extractor de características propuesto, permite mejorar notablemente los resultados del proceso de registración de imágenes respecto de los obtenidos con los dos algoritmos, muy conocidos en esta área, como son SIFT y SURF. El promedio de los valores de similaridad están en el rango $[0.82 - 0.95]$. Analizándolos con los obtenidos con SIFT y SURF, estos valores son mejores en el rango $[5\% - 8\%]$.

Estos resultados se deben a que la salida del filtro de Gabor imita el comportamiento de las células de la corteza visual, lo cual, permite comparar y combinar características para el proceso de registración de imágenes. Además, los momentos invariantes tienen propiedades matemáticas interesantes, como ortogonalidad y baja redundancia. Por tanto, los vectores descriptores permiten distinguir adecuadamente las regiones de cada marcador anatómico.

Respecto del tiempo de procesamiento, el algoritmo propuesto utiliza en promedio 10 segundos en el valor máximo de escala. Comparándolo con los otros dos algoritmos es el que mayor tiempo utiliza. Sin embargo, el promedio del tiempo es muy aceptable para aplicaciones que no requieran procesamiento en tiempo real.

Capítulo 5 - Inteligencia Computacional

Al utilizar algoritmos de aprendizaje por refuerzo, se indagó en un nuevo paradigma que no ha sido tan explotado para la registración de imágenes. Además, se abordó uno de los métodos más recientes en el caso del aprendizaje de máquina como es el aprendizaje profundo (*Deep Learning*), en el caso específico, del aprendizaje por refuerzo profundo.

De acuerdo con los resultados obtenidos, se puede concluir que estos algoritmos se adaptan bien al entorno de registración. Sin embargo, los requerimientos computacionales son mayores debido a las herramientas que utiliza como, por ejemplo, las redes neuronales. La propuesta de utilizar una memoria de respaldo para almacenar las mejores experiencias, bajo la arquitectura Doble DQN, permitió mejorar los resultados obtenidos por los demás algoritmos RL utilizados durante la comparación de rendimientos.

6.2 Conclusiones generales

La metodología seguida durante este trabajo ha tenido como base abordar los mejores métodos propuestos, junto con los recientemente presentados con el objetivo de brindar nuevas propuestas que estén enfocadas en mejorar los resultados obtenidos y aportar así una solución al problema de registración.

Estas propuestas están dadas en el campo de la ingeniería con base en matemática, estadística y las ciencias computacionales. Esto ha permitido cumplir con los objetivos planteados al inicio de esta investigación y, sin duda, abren la posibilidad para profundizar en esta área.

Dentro del procesamiento de imágenes y en las aplicaciones de ingeniería, en general, no existe un único método que se ajuste a todas las necesidades requeridas. Es por esto que la registración ha sido abordada desde tres perspectivas donde los resultados obtenidos aportan una solución diferente para el alineamiento óptimo de imágenes.

En el caso de los algoritmos de inteligencia colectiva y de enjambres, aportan una solución probabilística muy buena, además de utilizar eficientemente los recursos computacionales. El tiempo de procesamiento está muy ligado al número de iteraciones establecidas para la ejecución de estos algoritmos, por tanto, el usuario tiene el control directo de estos parámetros.

En el caso de la extracción de características locales es muy dependiente de la definición del espacio de escala y del rendimiento de los descriptores de textura, sin embargo, en el caso de las imágenes médicas los resultados obtenidos son los esperados incluso cuando se los compara con algoritmos muy conocidos. Además, el uso de los recursos computacionales es óptimo y el tiempo de procesamiento es mucho menor que el usado por el enfoque anterior.

En el caso de la inteligencia por refuerzo, requiere recursos computacionales muy buenos para que el proceso tenga un tiempo aceptable de procesamiento y los resultados puedan ser los esperados y generalizados. Pero sin duda los resultados que se pueden obtener van a ser mucho más precisos que en los dos casos anteriores.

Por tanto, no se puede concluir cuál de estos tres enfoques es mejor, ya que cada uno presenta ventajas y desventajas, lo cual, debe ser analizado de acuerdo al problema, el tiempo estimado de procesamiento y los recursos computacionales disponibles.

Se espera que estos aportes puedan tener un impacto a nivel social, especialmente regional, donde el equipamiento médico de alta tecnología aún no está disponible en todos los lugares. Estos algoritmos pueden servir para automatizar algunos procesos y de esta manera los especialistas médicos puedan disponer de una herramienta tecnológica que sirva de apoyo para la toma de decisiones de determinados procedimientos.

6.3 Trabajo futuro

La registración de imágenes es un proceso que abre la posibilidad de abordar varias temáticas apasionantes como son la optimización, la geometría computacional, la inteligencia computacional junto con el aprendizaje de máquina y, por supuesto, el procesamiento de imágenes. En cada una de estas áreas se pueden plantear trabajos de investigación, con la finalidad de realizar aportes que mejoren los algoritmos y procedimientos existentes.

Por tanto, este campo de investigación sigue siendo prometedor, ya que, el uso de las nuevas herramientas tecnológicas puede ser de gran utilidad para seguir mejorando los resultados actuales. El trabajo futuro se puede enfocar a la registración de imágenes en el dominio 3D y con transformaciones no lineales. Dentro de esta área es necesario contar con los recursos computacionales adecuados, lo cual, permitirá obtener un buen rendimiento de los algoritmos y métodos que se pueden aplicar.

Otra de las áreas prometedoras y con gran auge en los últimos años es el área del aprendizaje computacional utilizando los nuevos paradigmas como, por ejemplo, el Aprendizaje Profundo. Mediante el uso de estos paradigmas se puede profundizar en los métodos existentes, lo cual, permitirá aportar nuevos conocimientos que puedan mejorar los algoritmos y sus rendimientos actuales.