UNIVERSITAT POLITÈCNICA DE VALÈNCIA

_ TELECOM ESCUELA TÉCNICA VLC SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería de Telecomunicación

DIGICOR: Plataforma interactiva para la simulación de procedimientos de ablación en fibrilación auricular

Trabajo Fin de Grado

Grado en Ingeniería Física

AUTOR/A: Doallo Cercos, Elena

Tutor/a: Guillem Sánchez, María Salud

Director/a Experimental: Moreno López, Raúl

CURSO ACADÉMICO: 2024/2025

Resumen

La Fibrilación Auricular (FA) es la arritmia más común, y su tratamiento mediante ablación presenta tasas de recurrencia notables debido a la complejidad de sus mecanismos subyacentes. Este trabajo presenta DIGICOR, una plataforma de software interactiva diseñada para la simulación personalizada de la FA y la evaluación virtual de estrategias de ablación. Integrando un modelo de autómata celular C++/CUDA con una interfaz gráfica PyQt5/Vedo y módulos de análisis en MATLAB, DIGICOR permite cargar geometrías auriculares, configurar parámetros fisiológicos, simular la actividad eléctrica y aplicar lesiones de ablación de forma interactiva.

La plataforma facilita la visualización dinámica de la propagación del potencial de acción y la generación de mapas funcionales (Frecuencia Dominante, Estabilidad, Patrones Reentrantes) para caracterizar la arritmia. Una característica clave es la capacidad de guardar y comparar múltiples escenarios de ablación, permitiendo al usuario explorar diferentes enfoques terapéuticos. Los resultados experimentales con simulaciones de FA de diversa complejidad demuestran el potencial de DIGICOR para identificar mecanismos arrítmicos, guiar estrategias de ablación más dirigidas y potencialmente reducir la extensión de las lesiones en comparación con protocolos estándar, sirviendo como una prometedora herramienta de apoyo a la planificación clínica.

Resum

La Fibril·lació Auricular (FA) és l'arrítmia més comuna, i el seu tractament mitjançant ablació presenta taxes de recurrència notables a causa de la complexitat dels seus mecanismes subjacents. Este treball presenta DIGICOR, una plataforma de programació interactiva dissenyada per a la simulació personalitzada de la FA i l'avaluació virtual d'estratègies d'ablació. Integrant un model d'autòmat cel·lular C++/CUDA amb una interfície gràfica PyQt5/Vedo i mòduls d'anàlisis en MATLAB, DIGICOR permet carregar geometries auriculars, configurar paràmetres fisiològics, simular l'activitat elèctrica i aplicar lesions d'ablació de manera interactiva.

La plataforma facilita la visualització dinàmica de la propagació del potencial d'acció i la generació de mapes funcionals (Freqüència Dominant, Estabilitat, Patrons Reentrants) per a caracteritzar l'arrítmia. Una característica clau és la capacitat de guardar i comparar múltiples escenaris d'ablació, permetent a l'usuari explorar diferents enfocaments terapèutics. Els resultats experimentals amb simulacions de FA de diversa complexitat demostren el potencial de DIGICOR per a identificar mecanismes arrítmics, guiar estratègies d'ablació més dirigides i potencialment reduir l'extensió de les lesions en comparació amb protocols estàndard, servint com una prometedora ferramenta de suport a la planificació clínica.

Abstract

Atrial Fibrillation (AF) is the most common arrhythmia, and its treatment via ablation presents notable recurrence rates due to the complexity of its underlying mechanisms. This work introduces DIGICOR, an interactive software platform designed for the personalized simulation of AF and the virtual evaluation of ablation strategies. By integrating a C++/CUDA cellular automaton model with a PyQt5/Vedo graphical interface and MATLAB analysis modules, DIGICOR allows for the loading of atrial geometries, configuration of physiological parameters, simulation of electrical activity, and interactive application of ablation lesions.

The platform facilitates the dynamic visualization of action potential propagation and the generation of functional maps (Dominant Frequency, Stability, Reentrant Patterns) to characterize the arrhythmia. A key feature is the ability to save and compare multiple ablation scenarios, enabling the user to explore different therapeutic approaches. Experimental results with AF simulations of varying complexity demonstrate DIGICOR's potential to identify arrhythmic mechanisms, guide more targeted ablation strategies, and potentially reduce the extent of lesions compared to standard protocols, serving as a promising tool to support clinical planning.

Resumen ejecutivo

La memoria del TFG del Grado en Ingeniería Física debe desarrollar en el texto los siguientes conceptos, debidamente justificados y discutidos, centrados en el ámbito de la ingeniería física

CONCEPT (ABET)	CONCEPTO (traducción)	¿Cumple? (S/N)	¿Dónde? (páginas)
1. IDENTIFY:	1. IDENTIFICAR:		
1.1. Problem statement and opportunity	1.1. Planteamiento del problema y oportunidad	S	1-5
1.2. Constraints (standards, codes, needs, requirements & specifications)	1.2. Toma en consideración de los condicionantes (normas técnicas y regulación, necesidades, requisitos y especificaciones)	S	6-11
1.3. Setting of goals	1.3. Establecimiento de objetivos	S	12, 13
2. FORMULATE:	2. FORMULAR:		
2.1. Creative solution generation (analysis)	2.1. Generación de soluciones creativas (análisis)	S	14-27
2.2. Evaluation of multiple solutions and decision-making (synthesis)	2.2. Evaluación de múltiples soluciones y toma de decisiones (síntesis)	S	28-45
3. SOLVE:	3. RESOLVER:		
3.1. Fulfilment of goals	3.1. Evaluación del cumplimiento de objetivos	S	46-53
3.2. Overall impact and significance (contributions and practical recommendations)	3.2. Evaluación del impacto global y alcance (contribuciones y recomendaciones prácticas)	S	56-58

Escuela Técnica Superior de Ingeniería de Telecomunicación
Universitat Politècnica de València
Edificio 4D. Camino de Vera, s/n, 46022 Valencia
Tel. +34 96 387 71 90, ext. 77190
www.etsit.upv.es

Escuela Técnica Superior de Ingeniería de Telecomunicación
Universitat Politècnica de València
Edificio 4D. Camino de Vera, s/n, 46022 Valencia
Tel. +34 96 387 71 90, ext. 77190
www.etsit.upv.es

Lista de Tablas

Escuela Técnica Superior de Ingeniería de Telecomunicación
Universitat Politècnica de València
Edificio 4D. Camino de Vera, s/n, 46022 Valencia
Tel. +34 96 387 71 90, ext. 77190
www.etsit.upv.es

Escuela Técnica Superior de Ingeniería de Telecomunicación
Universitat Politècnica de València
Edificio 4D. Camino de Vera, s/n, 46022 Valencia
Tel. +34 96 387 71 90, ext. 77190
www.etsit.upv.es

Lista de Figuras

Escuela Técnica Superior de Ingeniería de Telecomunicación
Universitat Politècnica de València
Edificio 4D. Camino de Vera, s/n, 46022 Valencia
Tel. +34 96 387 71 90, ext. 77190
www.etsit.upv.es

Escuela Técnica Superior de Ingeniería de Telecomunicación
Universitat Politècnica de València
Edificio 4D. Camino de Vera, s/n, 46022 Valencia
Tel. +34 96 387 71 90, ext. 77190
www.etsit.upv.es

Escuela Técnica Superior de Ingeniería de Telecomunicación
Universitat Politècnica de València
Edificio 4D. Camino de Vera, s/n, 46022 Valencia
Tel. +34 96 387 71 90, ext. 77190
www.etsit.upv.es

Escuela Técnica Superior de Ingeniería de Telecomunicación
Universitat Politècnica de València
Edificio 4D. Camino de Vera, s/n, 46022 Valencia
Tel. +34 96 387 71 90, ext. 77190
www.etsit.upv.es

Tabla de Contenido

Escuela Técnica Superior de Ingeniería de Telecomunicación
Universitat Politècnica de València
Edificio 4D. Camino de Vera, s/n, 46022 Valencia
Tel. +34 96 387 71 90, ext. 77190
www.etsit.upv.es

Escuela Técnica Superior de Ingeniería de Telecomunicación
Universitat Politècnica de València
Edificio 4D. Camino de Vera, s/n, 46022 Valencia
Tel. +34 96 387 71 90, ext. 77190
www.etsit.upv.es

Escuela Técnica Superior de Ingeniería de Telecomunicación
Universitat Politècnica de València
Edificio 4D. Camino de Vera, s/n, 46022 Valencia
Tel. +34 96 387 71 90, ext. 77190
www.etsit.upv.es

Escuela Técnica Superior de Ingeniería de Telecomunicación
Universitat Politècnica de València
Edificio 4D. Camino de Vera, s/n, 46022 Valencia
Tel. +34 96 387 71 90, ext. 77190
www.etsit.upv.es

1. Introducción

1.1. Contexto médico

La fibrilación auricular (FA) es la arritmia cardíaca sostenida más frecuente en la práctica clínica y representa un desafío significativo para los sistemas sanitarios debido a su alta prevalencia, su asociación con complicaciones graves y la limitada eficacia de los tratamientos actuales [1]. Esta patología consiste en una activación eléctrica rápida, irregular y desorganizada de las aurículas, que conlleva una contracción mecánica ineficaz del tejido auricular [1]. Como resultado, se produce una estasis sanguínea que aumenta el riesgo de formación de trombos y, por ende, de eventos embólicos como los accidentes cerebrovasculares isquémicos [2].

1.1.1. Epidemiología y carga sanitaria

Según estimaciones recientes, más de 43 millones de personas en todo el mundo padecen FA [3]. Esta prevalencia ha mostrado un aumento constante, considerando que en 2010 el número de afectados se estimaba en 33.5 millones [4], y se proyecta que esta tendencia al alza continue, impulsada principalmente por el envejecimiento poblacional y el incremento de factores de riesgo como la hipertensión, la obesidad o la insuficiencia cardíaca [1]. La FA también muestra dependencia con la edad, en 2010 se estimó que en la Unión Europea 8.8 millones de los adultos que padecían FA eran mayores de 55 años y se proyecta que este número se duplique para el año 2060, alcanzando los 17.9 millones [5].

La FA tiene un impacto importante no solo a nivel clínico, sino también sanitario y económico [6]. Representa una de las principales causas de hospitalización por arritmias y se asocia a un incremento en la mortalidad, el deterioro funcional y la pérdida de calidad de vida, especialmente en pacientes de edad avanzada [6][7]. Se estima que cerca del 20–30 % de los ictus isquémicos en adultos mayores están relacionados con la FA [1], debido a la formación de trombos que ocurren principalmente en la orejuela izquierda [8]. A esto se suma que los pacientes con FA tienen mayor probabilidad de desarrollar o agravar insuficiencia cardíaca [1], y que la coexistencia de FA con IC se asocia con un riesgo de muerte más de dos veces superior [9].

En términos de costes, se calcula que la FA representa más del 1% del gasto sanitario total en países desarrollados, incluyendo tanto costes directos (hospitalizaciones, procedimientos médicos) como indirectos (pérdida de productividad, dependencia funcional) [10].

1.1.2. Mecanismos electrofisiológicos

Antes de profundizar en los complejos mecanismos electrofisiológicos que subyacen a la fibrilación auricular y en las estrategias computacionales para su estudio, es fundamental establecer un conocimiento básico de la anatomía cardíaca relevante. Una comprensión clara de la estructura general de las aurículas y sus regiones clave facilitará la interpretación de los conceptos y resultados presentados a lo largo de este documento.

Estructura anatómica de las aurículas

La Figura 1.1 presenta una geometría auricular muy similar a la utilizada como base para las simulaciones en este proyecto, mostrando diferentes perspectivas para una mejor apreciación

tridimensional. Las estructuras más importantes para el contexto de la fibrilación auricular, y que se referenciarán frecuentemente, incluyen:

- **Aurícula Izquierda (LA, left atrial) y Aurícula Derecha (RA, right atrial):** Las cámaras superiores del corazón donde se desarrolla la actividad eléctrica. La FA se manifiesta predominantemente en la LA.
- **Venas Pulmonares (PV):** Etiquetadas en la figura como **Venas Pulmonares Izquierdas (LPVs, left pulmonary veins)** y **Venas Pulmonares Derechas (RPVs, right pulmonary veins)**, son cruciales ya que sus uniones con la aurícula izquierda son el origen más común de los focos ectópicos que inician la FA.
- **Orejuela Auricular Izquierda (LAA, left atrial appendage):** Este apéndice de la AI es de particular interés clínico debido a su rol en la formación de trombos.
- **Orejuela Auricular Derecha (RAA, right atrial appendage):** El apéndice correspondiente de la aurícula derecha.
- **Vena Cava Superior (SVC, superior vena cava) e Inferior (IVC, inferior vena cava):** Principales venas que drenan a la aurícula derecha.
- **Pared Posterior de la Aurícula Izquierda (PW, posterior wall):** La región situada entre los grupos de venas pulmonares, a menudo implicada en el sustrato de la FA.

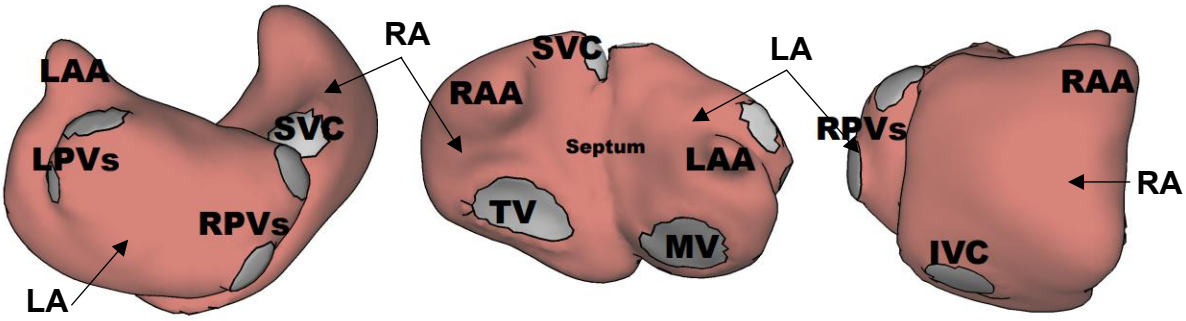


Figura 1.1 Geometría auricular. LPVs: left pulmonary veins (venas pulmonares izquierdas). RPVs: right pulmonary veins (venas pulmonares derechas). LAA: left atrial appendage (Orejuela auricular izquierda). RAA: right atrial appendage (Orejuela auricular derecha). SVC: superior vena cava. IVC: inferior vena cava. MV: mitral valve (válvula mitral). TV: tricuspid valve (válvula tricúspide).

Esta caracterización anatómica, basada en los modelos proporcionados por ITACA-COR (Universitat Politècnica de València) y Corify Care S.L. (Madrid, España), servirá de referencia para la discusión de los mecanismos electrofisiológicos de la FA y la interpretación de los resultados de simulación que se exponen a continuación.

Mecanismos electrofisiológicos

Desde el punto de vista electrofisiológico, la fibrilación auricular suele iniciarse por focos ectópicos o desencadenantes que generan impulsos rápidos en las aurículas. Estos focos se localizan predominantemente en las venas pulmonares, tal como identificó de forma pionera Haissaguerre y colaboradores en 1998 [11]. Otros sitios potenciales menos frecuentes incluyen la vena cava superior, el seno coronario o el ligamento de Marshall. Sin embargo, para que la arritmia se mantenga en el tiempo, no basta con un disparo inicial: debe existir un sustrato auricular susceptible que permita la perpetuación de la actividad eléctrica desorganizada [12].

En este contexto, se han propuesto diferentes mecanismos para explicar el mantenimiento de la FA, que no son mutuamente excluyentes. Entre ellos, destaca la teoría de las múltiples ondas de reentrada, formulada inicialmente por Moe [13] y desarrollada posteriormente por Allessie, según la cual numerosos frentes de onda se fraccionan y conducen de forma errática a través del miocardio auricular, generando un patrón de activación fibrilatorio sostenido [14]. Paralelamente, se ha planteado la existencia de focos ectópicos de activación automática, principalmente en las venas pulmonares, y la presencia de rotores, es decir, fuentes rotacionales estables que actúan como organizadores del caos eléctrico (Figura 1.2) [15][16].

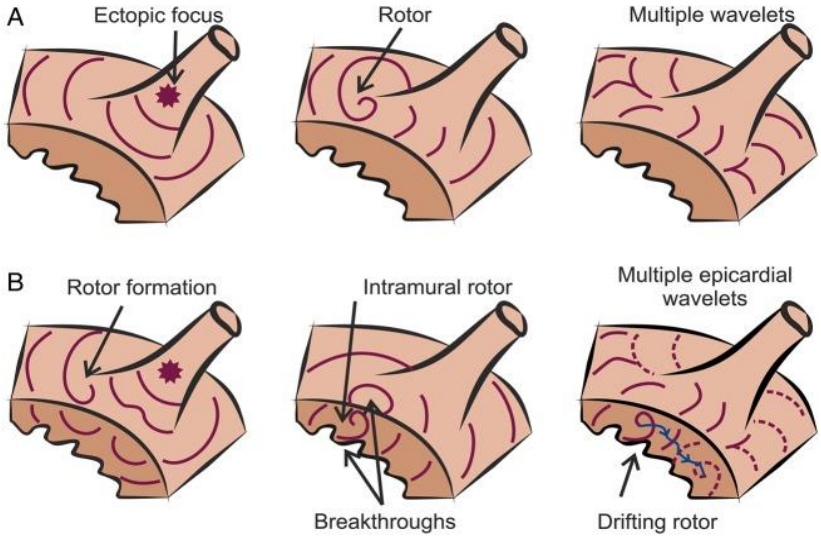


Figura 1.2 Hipótesis actual para el mantenimiento de la FA [16]. (A) Foco ectópico en la vena pulmonar. (B) Ejemplo de rotor. (C) Representación de múltiples wavelets. (D) Formación de rotor por un foco ectópico. (E) Puntos de penetración provocados por un rotor transmural. (F) Múltiples wavelets provocados por un rotor transmural errante (azul).

Con el tiempo, la propia FA produce modificaciones del sustrato que la favorecen, el remodelado eléctrico y el remodelado estructural. El eléctrico consiste en un acortamiento del período refractario y heterogeneidad en la conducción eléctrica, y el estructural puede ser tanto una dilatación de la aurícula, especialmente izquierda, como un engrosamiento de sus paredes y/o fibrosis del miocardio auricular que creando barreras de conducción y zonas de bloqueos de conducción facilitan circuitos de reentrada [17]. Investigaciones recientes han mostrado que algunos rotores tienden a localizarse en regiones anatómicas concretas, como la pared posterior de la aurícula izquierda o la orejuela, y que su comportamiento puede estar condicionado por la presencia de fibrosis subyacente [18].

En resumen, la FA se sostiene mediante la interacción de múltiples mecanismos: focos automáticos rápidos, reentradas funcionales y anatómicas; que dan lugar a una activación auricular caótica y descoordinada. Esta activación desorganizada conlleva una contracción mecánica ineficaz de las aurículas, generando estasis sanguínea que favorece la formación de trombos intracavitarios, con el consiguiente aumento del riesgo de eventos embólicos como el ictus isquémico [1][2].

1.1.3. Clasificación clínica

Desde un punto de vista clínico, la fibrilación auricular se clasifica según la duración de los episodios en tres formas: paroxística, persistente y permanente [1]. Esta categorización permite orientar el

manejo terapéutico y estimar el pronóstico, ya que refleja el grado de evolución del sustrato arrítmico y la probabilidad de recuperación del ritmo sinusal.

- **FA paroxística**: corresponde a episodios que terminan espontáneamente en menos de 7 días, habitualmente dentro de las primeras 24–48 horas. También se incluye en esta categoría la FA que es revertida mediante cardioversión dentro de ese mismo plazo.
- **FA persistente**: se refiere a episodios de duración superior a 7 días o que requieren intervención médica (cardioversión eléctrica o farmacológica) para su terminación. Cuando persiste durante más de 12 meses, se habla de FA persistente de larga duración. En estos casos, la reversión espontánea es poco probable, y suelen considerarse estrategias activas de control del ritmo o, alternativamente, de frecuencia si se descarta la restauración del ritmo sinusal.
- **FA permanente**: implica la aceptación clínica de la arritmia como estado crónico. Se establece cuando médico y paciente acuerdan no intentar más cardioversiones, ya sea por fracaso terapéutico previo o porque la arritmia se tolera adecuadamente.

Esta clasificación es útil para adaptar las decisiones terapéuticas, ya que la FA paroxística suele responder mejor a la ablación, mientras que la FA persistente y permanente suelen requerir enfoques más complejos o centrados en la estabilización hemodinámica y la prevención de complicaciones.

1.1.4. Estrategias de tratamiento actuales

El tratamiento de la fibrilación auricular se basa en tres pilares: control del ritmo, control de la frecuencia ventricular y prevención del tromboembolismo [1]. Además, el manejo de comorbilidades como hipertensión, insuficiencia cardíaca o apnea del sueño forma parte del enfoque terapéutico [19].

Control del ritmo. La restauración y mantenimiento del ritmo sinusal puede lograrse con fármacos antiarrítmicos (FAA) o mediante ablación por catéter. Aunque los FAA siguen siendo la primera opción en muchos casos, presentan eficacia limitada y frecuentes efectos adversos. En este contexto, la ablación transcatéter ha emergido como una estrategia más eficaz para reducir la carga arrítmica y mejorar la calidad de vida [19].

El procedimiento de ablación más frecuente se basa en el aislamiento eléctrico de las venas pulmonares (PVI, pulmonary vein isolation) (Figura 1.3), dado que estas suelen contener los focos ectópicos que inician la FA [11]. La técnica puede realizarse mediante radiofrecuencia punto a punto o crioablación con balón, y ambas han demostrado eficacia comparable en FA paroxística [19]. En FA persistente, donde el sustrato es más complejo, pueden añadirse lesiones adicionales (líneas en el techo, línea mitral y zonas de rotores), aunque su beneficio es discutido [20][21].

Figura 1.3 Ejemplo de las líneas de ablación de PVI. Obtenido de [22]

Control de la frecuencia. Cuando no se busca revertir la FA, se emplean fármacos como betabloqueantes. Los calcioantagonistas se usan con cautela, especialmente si hay insuficiencia cardíaca . La digoxina puede complementar, aunque no mejora la supervivencia y su seguridad en FA con insuficiencia cardiaca es debatida [23].

Anticoagulación. Todos los pacientes con riesgo embólico elevado deben recibir anticoagulación oral, independientemente de la estrategia elegida o del mantenimiento del ritmo sinusal [1].

Aunque estas estrategias han mejorado significativamente el manejo de la FA, su eficacia varía considerablemente según el tipo de paciente y la forma clínica de la arritmia. En especial, la FA persistente continúa presentando retos considerables, lo que ha impulsado la búsqueda de nuevas herramientas terapéuticas que permitan adaptar el tratamiento a las características individuales del sustrato auricular.

1.2. Relevancia del proyecto

Pese a los avances en técnicas de ablación, farmacología y dispositivos, las estrategias actuales para el tratamiento de la FA presentan limitaciones relevantes. La ablación por catéter basada en el aislamiento de las venas pulmonares continúa siendo el pilar fundamental del tratamiento, con resultados muy favorables en FA paroxística. Sin embargo, su eficacia disminuye notablemente en FA persistente, donde aproximadamente la mitad de los pacientes presentan recurrencias tras una única intervención [24].

Esta diferencia de resultados se explica por la complejidad del sustrato eléctrico y estructural subyacente, que no siempre se refleja en la anatomía auricular visible. En estos casos, el éxito del tratamiento dependería de una correcta identificación de las regiones críticas que mantienen la arritmia. Sin embargo, en la práctica clínica actual no existen criterios individualizados claramente definidos para orientar qué lesiones aplicar más allá del PVI. Las decisiones terapéuticas suelen basarse en protocolos anatómicos generalistas o en la experiencia del operador, lo que genera una notable variabilidad en los resultados.

Además, las herramientas disponibles como el mapeo electroanatómico invasivo, aunque permiten generar mapas tridimensionales de activación y voltaje al desplazar un catéter por la aurícula, son técnicamente exigentes: prolongan la intervención al requerir un muestreo secuencial de alta densidad que solo ofrece información local y fragmentaria, dependen en gran medida de la destreza del operador y nunca capturan de forma simultánea la actividad global de toda la cavidad, lo que puede pasar por alto mecanismos transitorios o sustratos sutiles que perpetúan la FA [25]. Todo ello

pone de manifiesto la necesidad urgente de desarrollar nuevas tecnologías que permitan una planificación terapéutica más precisa y guiada por información funcional específica del paciente. DIGICOR nace precisamente para llenar ese vacío, presentándose como una aplicación basada en ECGI no invasiva y modelado electroanatómico personalizado, ofrece un mapa global y dinámico de la aurícula para optimizar la estrategia de ablación.

1.3. Estado del arte

El desarrollo de tecnologías para caracterizar de forma no invasiva el sustrato eléctrico de la aurícula durante la fibrilación auricular ha impulsado una línea de investigación muy activa. En particular, se han producido avances relevantes en tres áreas clave: la imagen electrocardiográfica (ECGI), los modelos computacionales del corazón y los gemelos digitales. Estos componentes han dado lugar a herramientas experimentales y plataformas de simulación que, aunque prometedoras, todavía presentan limitaciones importantes para su adopción clínica generalizada. En este apartado se revisan las principales tecnologías existentes y se contextualiza el enfoque del proyecto DIGICOR dentro del panorama actual.

1.3.1. Mapeo Electrocardiográfico de Superficie Corporal (BSPM) e Imagen Electrocardiográfica (ECGI)

La caracterización no invasiva de la FA ha avanzado significativamente gracias al **Mapeo Electrocardiográfico de Superficie Corporal (BSPM, Body Surface Potential Mapping)**. A diferencia del electrocardiograma (ECG) estándar de 12 derivaciones, el BSPM utiliza un mayor número de electrodos (típicamente entre 32 y 256) distribuidos sobre una amplia área del torso para capturar con mayor detalle la distribución espacial de los potenciales eléctricos generados por el corazón (Figura 1.4) [26]. Si bien el ECG tradicional ha dominado la práctica clínica, estudios en las últimas décadas han demostrado que el BSPM contiene información clínicamente relevante adicional [27], especialmente en arritmias complejas como la FA, donde múltiples frentes de onda generan patrones eléctricos variables en el torso [28].

El BSPM es el dato de entrada fundamental para la **ECGI**, una tecnología que reconstruye la actividad eléctrica directamente sobre la superficie del corazón. Tradicionalmente, los modelos anatómicos para ECGI se obtenían mediante imagen médica. Sin embargo, para superar las limitaciones de coste, radiación y tiempo asociadas, se han desarrollado y validado enfoques de **ECGI "imageless"** [29]. Estos métodos utilizan técnicas como la fotogrametría (escaneo 3D) para adquirir la geometría del torso y luego estiman la geometría y posición del corazón, permitiendo una aplicación más ágil en la práctica clínica.

En el contexto de la ECGI "imageless", un ejemplo destacado de su implementación clínica y desarrollo es el trabajo realizado por el grupo ITACA-COR en colaboración con Corify Care S.L. [30], que ha resultado en el sistema **Acorys**. Este sistema utiliza fotogrametría para capturar la geometría del torso del paciente y adquiere los potenciales de superficie corporal con un chaleco de 128 electrodos. Posteriormente, la geometría de las aurículas se estima en relación con la forma del torso para resolver el problema inverso. El flujo de trabajo de **Acorys** está diseñado para integrarse en la práctica clínica, reduciendo costes, eliminando la exposición a radiación ionizante y disminuyendo el tiempo de preparación del paciente. El sistema genera mapas que permiten analizar patrones de activación y ha sido validado en diversos estudios para la estratificación de pacientes y la planificación de la ablación [26].

Asimismo, las señales epicárdicas reconstruidas por ECGI permiten derivar mapas funcionales como los mapas de frecuencia dominante (DF) que han sido utilizados para localizar regiones de actividad rápida o rotacional, candidatas a ablación [31]. Estas métricas funcionales han facilitado la identificación no invasiva de zonas clave en estudios preclínicos y en entornos experimentales.

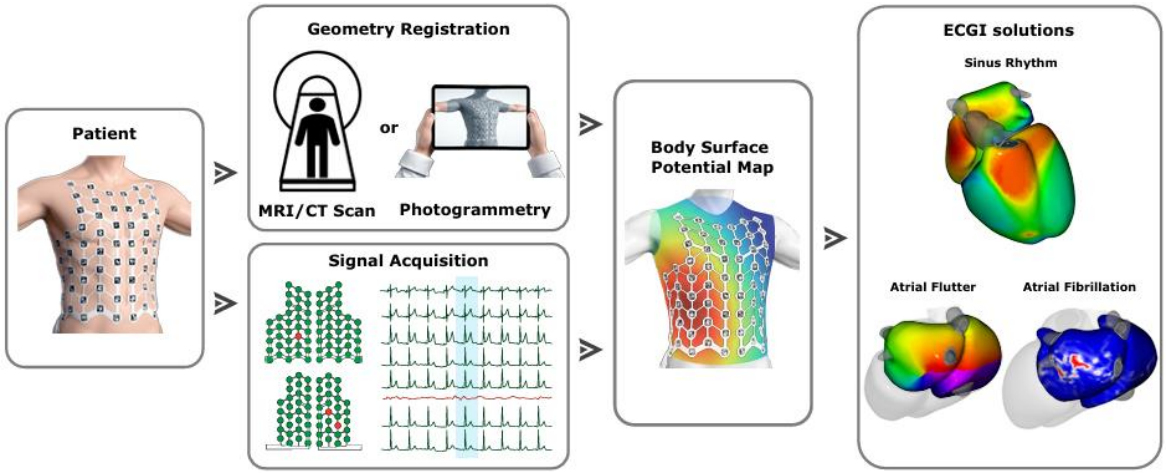


Figura 1.4 Flujo de trabajo de un estudio de ECGI realizado en las aurículas. El proceso comienza con la colocación de un chaleco de electrodos en el paciente para registrar las señales. Paralelamente, se realiza una resonancia magnética (RM), una tomografía computarizada (TC) o una fotogrametría. Esto permite construir un modelo geométrico tanto del corazón como del torso. Posteriormente, se utiliza el mapeo del potencial de superficie corporal para registrar las señales eléctricas del torso. Estas señales se utilizan para resolver el problema inverso de la electrocardiografía. Finalmente, se visualiza la actividad eléctrica de las aurículas mediante un modelo 3D realista y personalizado. De [26]

La capacidad de sistemas como Acorys para generar estos mapas epicárdicos a partir de mediciones en el torso se basa en la resolución de un desafío fundamental: el **problema inverso de la electrocardiografía**. Este problema busca determinar las fuentes eléctricas cardíacas a partir de sus manifestaciones en la superficie corporal.

Problema inverso

Para comprender el problema inverso, es útil primero considerar el **problema directo (forward problem)** de la electrocardiografía. Este describe cómo las fuentes eléctricas dentro del corazón (x) generan los potenciales eléctricos medibles en la superficie del torso (y) [32]. Debido a la complejidad matemática del modelo adaptado a dominios anatómicos realistas, se emplean estrategias de discretización para transformar la formulación analítica (basada en ecuaciones diferenciales) en un sistema lineal. En este sistema lineal, tanto la solución (potenciales del torso) como la información de entrada (fuentes cardíacas) se representan como vectores que contienen valores espaciales discretos. La relación entre las fuentes cardíacas y el voltaje del torso se encapsula en una matriz A, conocida como el operador directo o matriz de transferencia, que resulta de la discretización del modelo físico y geométrico. Incluyendo el ruido de medición (n), esta relación se puede expresar linealmente como:

$$y = A \cdot x + n \tag{1.1}$$

El cálculo de la matriz de transferencia A a menudo se basa en métodos como el Método de Elementos de Contorno (BEM, Boundary Element Method) es una forma de resolver problemas físicos complejos centrándose solo en los bordes o superficies de los objetos, en este caso, las superficies externas del torso y el corazón. Esto simplifica los cálculos porque se trabaja con menos puntos [32].

El **problema inverso de la electrocardiografía**, por lo tanto, busca estimar los potenciales epicárdicos (x) a partir de los potenciales registrados en la superficie del torso (y), utilizando la matriz de transferencia A previamente calculada [33]. Sin embargo, la complejidad del problema inverso radica en que es inherentemente **mal condicionado (ill-posed)** [33]. La matriz de transferencia A, aunque bien definida para el problema directo, a menudo está mal condicionada para su inversión. Esto significa que no siempre existe una solución única para x y que la solución es extremadamente sensible a pequeñas perturbaciones**,** mínimos errores en los potenciales medidos en el torso o imprecisiones en la información geométrica pueden traducirse en alteraciones muy grandes y fisiológicamente irreales en los potenciales epicárdicos reconstruidos (x).

Debido a este mal condicionamiento, no es posible obtener una solución estable y realista de forma inmediata, asumiendo que A fuera invertible, lo cual no siempre es el caso. Por ello, se necesitan aplicar técnicas de regularización matemática. Estas técnicas introducen restricciones adicionales o conocimiento previo sobre la naturaleza de la solución esperada x para estabilizar el problema y obtener resultados fisiológicamente plausibles.

Una de las técnicas de regularización más comúnmente empleadas es la **regularización de Tikhonov**. La regularización de Tikhonov busca encontrar el vector x que minimiza la siguiente función objetivo [34]:

$$\hat{x} = argmin\,\{\|y - A \cdot x\|_2^2 + \lambda^2 \|L \cdot x\|_2^2\} \tag{1.2}$$

El primer término cuantifica el error de ajuste a los datos medidos y el segundo impone una penalización a la "complejidad" o "energía" de la propia solución cardíaca. Estos términos están ponderados por un parámetro de regularización λ, que controla la intensidad de la penalización. El operador L es la matriz de regularización; para la regularización de Tikhonov de orden cero, L es la matriz identidad (I). El resultado de la minimización es:

$$\hat{x} = (A^T A + \lambda^2 L^T L)^{-1} A^T y \tag{1.3}$$

Donde $\hat{x}$ es el vector estimado de los potenciales epicárdicos (solución del problema inverso regularizado). Para resolverlo se debe determinar el valor adecuado para λ, que se puede obtener por diferentes métodos, uno de los más empleados es la curva L (L-curve) que consiste en resolver el problema regularizado para un rango de valores de λ y luego graficar frente al término residual [35]. La curva resultante tiene forma de L, donde el codo representa el equilibrio ideal entre fidelidad a los datos y la regularidad de la solución.

En respuesta a los desafíos y limitaciones asociados con el uso rutinario de imagen médica (coste, radiación, tiempo), han surgido y se han validado enfoques de ECGI "imageless" (sin imagen médica). Estos métodos buscan obtener la geometría del torso mediante técnicas alternativas, como la fotogrametría (escaneo 3D de la superficie corporal), y luego estimar la geometría y posición del corazón basándose en la forma del torso y utilizando bases de datos anatómicas o modelos estadísticos de forma. El trabajo de Molero et al. [26] exploró la viabilidad de la ECGI "imageless",

demostrando que, incluso con ciertas incertidumbres en la localización o geometría auricular estimada, se puede obtener una estimación robusta de parámetros electrofisiológicos cardíacos.

1.3.2. Modelos computacionales del corazón

Los modelos bioeléctricos permiten simular la propagación de la actividad eléctrica en el tejido cardíaco, desde un nivel celular hasta el órgano completo. Existen aproximaciones y modelos simplificados como los autómatas celulares (AC). Estos modelos han permitido estudiar cómo afectan la fibrosis, la heterogeneidad de conducción o las alteraciones iónicas al mantenimiento de la FA.

Los modelos de AC de la actividad eléctrica en la superficie del corazón son una representación simplificada pero potente de la electrofisiología cardíaca. Los AC describen de manera intuitiva cómo las células cardíacas se activan (despolarizan) y desactivan (repolarizan) mediante reglas de actualización sencillas para el estado de cada célula individual.

El grupo ITACA-COR ha aprovechado la tecnología "imageless" para avanzar en sus entornos de simulación personalizados basados en autómatas celulares. Los modelos de autómata se ajustan a las geometrías auriculares que son generadas mediante la técnica de ECGI "imageless" proporcionada por el sistema **Acorys**. Las mallas auriculares resultantes, compuestas por miles de nodos, sirven de base para el modelo de autómata celular, en el cual se pueden ajustar parámetros clave como la difusividad (que simula la velocidad de conducción) o la duración del potencial de acción. Estos modelos computacionales personalizados, construidos sobre geometrías "imageless", han permitido al grupo explorar virtualmente el efecto de diferentes estrategias de ablación y analizar su impacto sobre la dinámica eléctrica auricular.

1.3.3. Gemelos digitales (Digital Twins)

Los gemelos digitales cardíacos son modelos computacionales ajustados a los datos específicos de un paciente. Combinan la geometría auricular, mapas funcionales, datos eléctricos y modelos de conducción para simular la FA y predecir el efecto de distintas estrategias de ablación.

Un aspecto fundamental en la creación de gemelos digitales cardíacos es la obtención de una geometría auricular precisa y personalizada. Si bien la segmentación de imágenes médicas es una vía, una metodología avanzada empleada por el grupo ITACA-COR y Corify Care S.L. implica el uso de Modelos Estadísticos de Forma (SSM, statistical shape model). Los SSM son modelos geométricos que ofrecen una representación compacta de un conjunto de objetos tridimensionales semánticamente similares, capturando tanto su forma promedio como las principales variaciones observadas en una población [36]. A partir de estos SSM, se pueden generar diversas geometrías auriculares y torácicas que representan un espectro realista de la anatomía. En sistemas como Acorys, y en la visión de DIGICOR para la creación de auténticos gemelos digitales, se selecciona o adapta una geometría de este atlas estadístico para que se ajuste óptimamente a las características no invasivas de un paciente específico.

En esta línea se enmarca DIGICOR, una plataforma experimental diseñada para operar sobre geometrías auriculares específicas y con capacidad de simular de forma casi instantánea el efecto de diferentes lesiones ablativas a partir de datos electrofisiológicos no invasivos.

1.3.4. Comparación con plataformas existentes

Diversas herramientas de simulación y análisis se han propuesto como soporte al estudio o tratamiento de la FA. A continuación, se resumen las más relevantes, señalando sus aportes y limitaciones frente a la propuesta de DIGICOR:

- **openCARP**: plataforma de simulación biofísica de código abierto muy potente en investigación, pero sin capacidad de simulación interactiva en tiempo real ni interfaz accesible para uso clínico [37].
- **Sim4Life**: entorno comercial multifísico que permite simular conducción eléctrica y entrega de energía. Aunque versátil, no está específicamente diseñado para FA ni para uso por personal clínico. Requiere configuración manual compleja y tiempo prolongado por paciente [38].
- **OpenEP**: orientado al análisis post-procedimiento de datos electroanatómicos, no a simulación predictiva. Muy útil en investigación y validación de métricas, pero no simula dinámicamente la evolución de la arritmia [39].

Aunque estas herramientas han contribuido significativamente al conocimiento de la FA y al desarrollo de nuevas métricas, persiste una brecha en cuanto a plataformas que combinen la simulación biofísica predictiva con una alta interactividad en tiempo real y una interfaz diseñada específicamente para el entorno clínico. DIGICOR se diferencia al proponer una plataforma accesible y rápida para que el personal clínico e investigador pueda explorar y personalizar virtualmente estrategias de ablación, llenando un vacío en las herramientas de apoyo a la decisión disponibles actualmente para el tratamiento de esta arritmia.

1.3.5. Contribución diferencial de DIGICOR

Para abordar las necesidades específicas de la planificación de ablaciones de FA en la práctica clínica, DIGICOR se ha diseñado con un conjunto de características diferenciales. Estas capacidades buscan ofrecer una herramienta que combine la simulación predictiva con la interactividad y la personalización, superando algunas de las limitaciones de las plataformas existentes:

- Permite adaptar los parámetros eléctricos del modelo auricular a las características funcionales del paciente, optimizando valores como la difusividad, la frecuencia dominante y la variabilidad del intervalo de activación. Esta capacidad de personalización permite representar con mayor fidelidad el sustrato fibrilatorio y explorar su comportamiento bajo distintas condiciones, optimizando así la simulación y la planificación terapéutica.
- Reconstrucción de mapas funcionales como a partir de señales ECGI. Se generan métricas derivadas, como mapas de **frecuencia dominante media**, **mapas de estabilidad** o **mapas de patrones de reentrada,** que permiten caracterizar el sustrato fibrilatorio y localizar regiones potencialmente críticas para la perpetuación de la FA
- Simulación interactiva de estrategias de ablación. Durante la simulación de la activación eléctrica del tejido auricular, el usuario puede aplicar lesiones virtuales directamente sobre la geometría, observando de inmediato cómo varía la propagación del impulso eléctrico y comparando distintas estrategias de ablación. Esta funcionalidad permite evaluar el impacto de cada configuración en tiempo casi real, facilitando la identificación de dianas terapéuticas y optimizando la planificación del procedimiento.

DIGICOR representa un paso hacia la integración de la simulación en la toma de decisiones terapéuticas, facilitando un abordaje verdaderamente personalizado de la fibrilación auricular.

1.4. Motivación

La elevada tasa de recurrencias tras la ablación (30–50 % en FA persistente), junto con la falta de herramientas que integren información anatómica y funcional en tiempo real, subraya la necesidad de nuevas soluciones tecnológica.

DIGICOR surge como una respuesta directa, incorpora un núcleo de simulación basado en un autómata celular ajustado a geometrías auriculares que ha sido previamente utilizado para reproducir patrones eléctricos complejos y explorar distintas estrategias de ablación en estudios previos. La integración de este motor en una interfaz interactiva como DIGICOR abre la puerta a una planificación terapéutica más precisa, adaptada a las características funcionales de cada paciente.

Su implementación en la práctica clínica podría mejorar la planificación de los procedimientos, personalizar las decisiones terapéuticas en función del comportamiento eléctrico de cada paciente y evitar intervenciones innecesarias. Asimismo, permitiría reducir el tiempo de intervención, minimizar riesgos asociados al procedimiento y, en última instancia, mejorar los resultados clínicos y la calidad de vida del paciente. A largo plazo, su arquitectura modular ofrece posibilidades de expansión hacia otras arritmias auriculares y su integración con técnicas de inteligencia artificial refuerza su potencial como herramienta avanzada en el marco de la medicina personalizada.

2. Objetivos

2.1. Objetivo general

El objetivo general de este trabajo es desarrollar la plataforma interactiva DIGICOR para la simulación y evaluación de estrategias de ablación personalizadas en pacientes con fibrilación auricular, integrando modelos computacionales, visualización 3D y herramientas de análisis funcional que permitan representar de forma rápida e intuitiva la respuesta eléctrica del tejido auricular ante distintas intervenciones terapéuticas.

2.2. Objetivos específicos

- **Reproducir el flujo de trabajo existente de simulación y análisis como punto de partida para el desarrollo de la plataforma.**
    - Integrar el modelo de autómata celular (AutomataCUDAv1.dll) para la ejecución de simulaciones.
    - Establecer la comunicación con MATLAB para la generación de parámetros fisiológicos y mapas funcionales.
- **Desarrollo del frontend de la aplicación DIGICOR**
    - Diseñar una interfaz gráfica intuitiva para la visualización de la geometría auricular y los resultados de simulación.
    - Implementar herramientas interactivas como el pincel de ablación, carga de máscaras externas y visualización de mapas.
- **Desarrollo del backend de la aplicación**
    - Implementar la lógica de control, gestión de simulación, conexión con la DLL y tratamiento de datos.
    - Establecer la comunicación con MATLAB para generar mapas espectrales y parámetros personalizados.
    - Gestionar los buffers de animación, historial de fotogramas y sincronización con la interfaz gráfica.
- **Integración del sistema y pruebas funcionales**
    - Unificar el frontend y backend en una arquitectura modular y eficiente.
    - Asegurar la comunicación fluida entre todos los componentes del sistema (Python–MATLAB–DLL).
    - Realizar pruebas de funcionalidad, estabilidad y sincronización.
- **Evaluación de estrategias de ablación**
    - Aplicar estrategias interactivas o predefinidas sobre la geometría auricular.
    - Analizar el efecto de dichas estrategias mediante mapas funcionales y métricas como frecuencia dominante o porcentaje de tejido activo.
    - Visualizar la evolución eléctrica post-ablación y validar el impacto de las lesiones.

- **Optimización y mejora del sistema**
    - Corregir errores y mejorar la velocidad de ejecución, respuesta de la interfaz y visualización.
    - Ajustar el diseño gráfico y la estructura de la interfaz para mayor usabilidad.
    - Realizar un análisis crítico sobre las fortalezas y limitaciones de DIGICOR.
- **Documentación de la solución implementada.**
    - Elaborar la documentación técnica de la arquitectura del software, detallando los principales componentes y el flujo de datos.
    - Crear una guía de usuario que describa las funcionalidades clave de la plataforma, incluyendo la configuración de simulaciones, la aplicación de ablaciones y la generación de mapas.

3. Metodología

La plataforma DIGICOR ha sido concebida para proporcionar un entorno computacional integral que permita el estudio personalizado de la fibrilación auricular. El diseño y desarrollo de sus funcionalidades se ha guiado por un flujo de trabajo centrado en el usuario, que abarca desde la preparación de modelos específicos del paciente hasta la simulación de la arritmia, el análisis de sus mecanismos y la evaluación virtual de estrategias terapéuticas. La Figura 3.1 ilustra este flujo conceptual de uso de la aplicación, que sirve como hoja de ruta para la metodología descrita en esta sección.

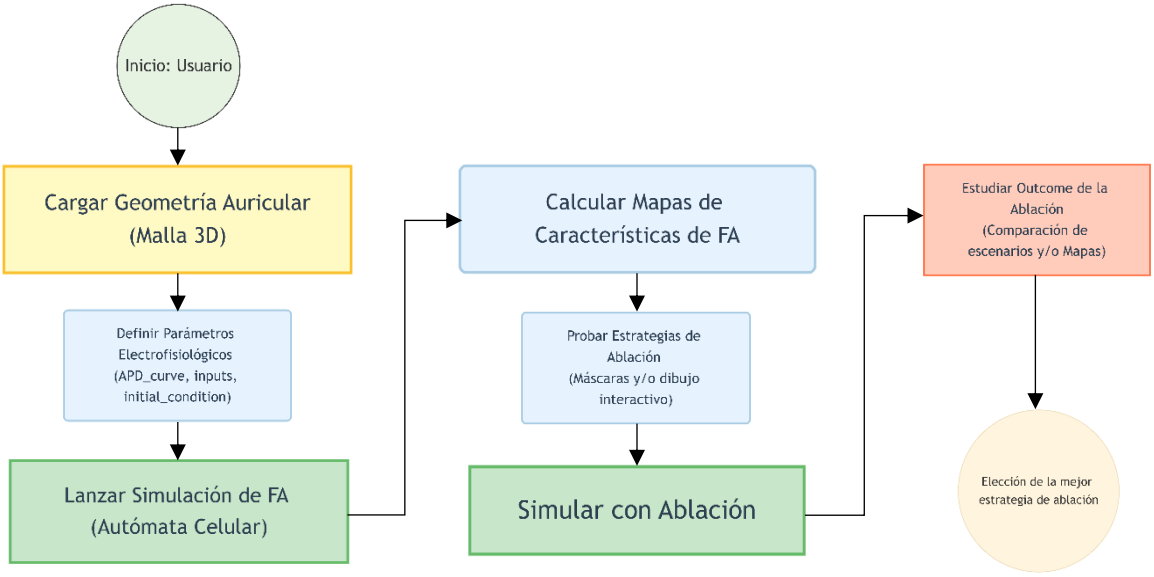


Figura 3.1 Idea conceptual de la aplicación DIGICOR

A continuación, se detallan los enfoques metodológicos adoptados para cada uno de estos componentes y funcionalidades clave de la plataforma DIGICOR.

3.1. Entorno de desarrollo

La implementación de DIGICOR se ha llevado a cabo en el entorno del grupo de investigación ITACA-COR, adscrito al Instituto ITACA de la Universitat Politècnica de València. Este grupo se dedica al problema inverso de la electrocardiografía y, dentro de ese marco, aborda diversas líneas de trabajo: el modelado eléctrico del corazón, la reconstrucción no invasiva de su actividad y el guiado y planificación de procedimientos de ablación, combinando así investigación en ingeniería biomédica con transferencia tecnológica.

En este contexto, DIGICOR se apoya en múltiples desarrollos previos del grupo, que han permitido construir una base robusta sobre la que avanzar en términos de interactividad, visualización y flexibilidad. Entre estos recursos destacan:

- La biblioteca AutomataCUDAv1.dll, que contiene el modelo de autómata celular tridimensional optimizado mediante CUDA para la simulación de la propagación eléctrica en el tejido auricular.

- Los scripts de MATLAB utilizados para generar las curvas de duración del potencial de acción y las condiciones iniciales personalizadas, esenciales para definir las propiedades electrofisiológicas del sustrato.
- Funciones validadas para el cálculo de mapas funcionales como la frecuencia dominante, estabilidad de la FA y el histograma de patrón reentrante, ya empleados en trabajos anteriores del grupo.

El trabajo desarrollado en este proyecto se ha centrado en extender y unificar estas herramientas en una plataforma visual, modular e interactiva, accesible a usuarios no expertos y orientada a futuras aplicaciones clínicas.

3.2. Planificación y organización del desarrollo

El desarrollo de DIGICOR se llevó a cabo siguiendo una metodología iterativa, estructurada en fases funcionales bien definidas y con un enfoque progresivo hacia la implementación de una herramienta interactiva, precisa y visualmente intuitiva para la simulación de fibrilación auricular y la exploración de estrategias de ablación personalizadas.

Desde el inicio, se estableció una organización basada en tres pilares fundamentales:

- **Seguimiento continuo semanal**, mediante un registro estructurado de tareas, avances, problemas encontrados y soluciones aplicadas.
- **Revisiones mensuales tipo one-to-one**, que sirvieron como puntos de control para evaluar el estado del desarrollo, redefinir objetivos a corto plazo y garantizar la alineación técnica con los objetivos de la herramienta.
- **Planificación funcional por bloques**, en la que cada etapa del desarrollo se centró en un componente clave del sistema: entorno de ejecución, visualización 3D, interacción con el usuario, conexión con el motor de simulación y herramientas de análisis.

Esta estructura organizativa permitió mantener una visión global del proyecto, facilitar la toma de decisiones técnicas y asegurar una evolución coherente del software, desde los primeros prototipos hasta una versión funcional completa.

Fases del desarrollo

La planificación se dividió en ocho fases funcionales, descritas a continuación y representadas en la Tabla 3.1, que resume el cronograma seguido durante el proceso de desarrollo.

Nombre de la tarea	Feb-25		Mar-25				Apr-25				May-25					Jun-25		
	11–17	18–24	25–03	04–10	11–17	18–24	25–31	01–07	08–14	15–21	22–28	29–05	06–12	13–19	20–26	27–02	03–09	10–16
1. Configuración del entorno□																		
1.1 Instalar dependencias necesarias																		
1.2 Verificar compatibilidad de versiones																		
1.3 Integrar MATLAB Engine API con Python																		
1.4 Probar importación de la DLL con ctypes																		
2. Diseño del backend y visualización preliminar																		
2.1 Diseñar esquema del flujo productor-consumidor																		
2.2 Implementar clases base para gestión de eventos y datos																		
2.3 Crear estructura para cargar geometrías																		
2.4 Prototipar la visualización 3D inicial con Vedo																		
2.5 Sincronizar flujo de datos entre productor y consumidor																		
2.6 Implementar la animación de potencial eléctrico en la geometría																		
2.7 Validar visualización básica con datos de prueba.																		
3. Definición de la interfaz gráfica																		
3.1 Esquematizar diseño general de la GUI																		
3.2 Implementar ventana principal y menús																		
3.3 Añadir panel lateral con pestañas o botones de navegación																		
3.4 Implementar controles de navegación																		
3.5 Probar interacciones básicas y ajustar diseño																		
4. Conexión con el motor de simulación																		
4.1 Diseñar estructuras de datos para enviar parámetros a la DLL																		
4.2 Implementar exportación de archivos de entrada desde MATLAB																		
4.3 Realizar simulaciones de prueba y depurar errores de conexión																		
5. Implementación de visualización dinámica																		
5.1 Sincronizar la animación con los controles temporales																		
5.2 Añadir control de velocidad, pausa y retroceso/avance																		
5.3 Validar funcionamiento																		
6. Herramientas de interacción y generación de mapas																		
6.1 Implementar modo dibujo con botón derecho del ratón																		
6.2 Cargar y aplicar máscaras externas sobre la geometría																		
6.3 Establecer comunicación con scripts MATLAB para cálculos																		
6.4 Visualizar mapas funcionales																		
6.5 Añadir opción para guardar las regiones dibujadas																		
7. Diseño del sistema de evaluación de estrategias																		
7.1 Implementar panel 'preview' para escenarios de ablación																		
7.2 Establecer un visor comparativo																		
8. Pruebas de consistencia y validación técnica																		
8.1 Evaluar la concordancia entre simulaciones ejecutadas en la plataforma y registros de simulaciones de referencia																		
8.2 Corrección de errores																		
8.3 Valoración de mejoras futuras																		

Tabla 3.1 Diagrama de Gantt del desarrollo de DIGICOR.

1. **Configuración del entorno (Febrero)**: Preparación del ecosistema de desarrollo: instalación y verificación de compatibilidad entre bibliotecas de visualización (VTK, Vedo), PyQt5 para la interfaz, integración con MATLAB y vinculación con la DLL en C++ mediante ctypes.
2. **Diseño del backend y visualización preliminar (Febrero – Marzo)**: Construcción del flujo interno basado en un modelo de productor-consumidor, encargado de la gestión del flujo de datos entre la simulación y la visualización. Simultáneamente, se generó una visualización básica de geometrías tridimensionales con Vedo, cargando archivos .mat
3. **Definición de la interfaz gráfica (Marzo)**: Diseño de los elementos principales de interacción: ventanas, menús, paneles laterales y controles para la navegación temporal.
4. **Conexión con el motor de simulación (Marzo – Abril)**: Integración funcional con la DLL AutomataCUDAv1.dll, estableciendo las funciones clave de simulación y la generación de archivos de entrada fisiológica a través de MATLAB.
5. **Implementación de visualización dinámica (Abril)**: Animación de datos eléctricos en tiempo real sobre la geometría visual, gestión de buffers circulares y mecanismos de control de reproducción.
6. **Herramientas de interacción y generación de mapas (Abril)**: Desarrollo del sistema de dibujo interactivo para definir ablaciones, carga de máscaras externas, y conexión con scripts de MATLAB para el cálculo de mapas funcionales.
7. **Diseño del sistema de evaluación de estrategias (Abril)**: Definición del sistema de métricas visuales y cuantitativas, orientadas a valorar el efecto de las estrategias aplicadas sobre la dinámica eléctrica simulada.
8. **Pruebas de consistencia y validación técnica (Mayo – Junio)**: Verificación de estabilidad, detección de errores y revisión de comportamiento en casos límite, con el objetivo de consolidar el funcionamiento general del sistema.

3.3. Arquitectura general de la plataforma

Para abordar la complejidad de una simulación interactiva de FA con visualización 3D y análisis de datos, se adoptó como metodología un diseño arquitectónico modular inspirado en el patrón Modelo-Vista-Controlador (MVC). Esta elección se fundamentó en la necesidad de separar las responsabilidades de la lógica de negocio (simulación y datos), la presentación al usuario, y el control de la interacción, con el fin de mejorar la mantenibilidad, escalabilidad y la posibilidad de desarrollo paralelo de los componentes.

Se diseñó un componente **Modelo** para encapsular toda la lógica de la simulación cardíaca y la gestión de los datos asociados. Metodológicamente, esto implicó la necesidad de interactuar con un motor de cálculo externo de alto rendimiento (la DLL C++/CUDA) y gestionar de forma eficiente grandes volúmenes de datos temporales y datos de configuración (parámetros, geometrías, máscaras). Se previó también la integración con un sistema externo (MATLAB) para tareas especializadas de generación de datos de entrada y análisis funcional.

El diseño del componente **Vista** se centró en proporcionar una representación visual 3D interactiva y rica en información, junto con una interfaz gráfica de usuario (GUI) intuitiva para el control y la exploración de datos. Se seleccionó PyQt5 como framework para la GUI y se planeó la integración de bibliotecas de visualización 3D potentes como Vedo/VTK.

Se concibió un componente Controlador para orquestar la interacción entre el Usuario, la Vista y el Modelo. Su rol metodológico sería el de interpretar las acciones del usuario provenientes de la Vista, traducirlas en comandos para el Modelo, y asegurar que la Vista refleje consistentemente el estado actual del Modelo. Se anticipó la necesidad de gestionar la comunicación asíncrona para tareas computacionalmente intensivas, utilizando el sistema de señales y slots de PyQt5 y la ejecución en hilos**.**

La cohesión entre estos componentes se planificó mediante un sistema de gestión de eventos y un flujo de datos bien definido (ver Figura 3.2 para una representación esquemática de este flujo de datos general diseñado) para asegurar la interactividad y eficiencia.

3.3.1 Flujo de datos

Para garantizar la trazabilidad interna de DIGICOR y facilitar la depuración, a continuación, se describe de forma visual (Figura. 3.2) y paso a paso cómo se diseñó el flujo de datos desde la acción del usuario hasta la visualización final de la simulación y los análisis posteriores.

1. Entrada del usuario

El proceso comienza en el "Entorno del Usuario", donde el usuario carga archivos, define parámetros fisiológicos y de simulación mediante los diálogos de la GUI y escoge la geometría sobre la que trabajar.

2. Control principal & GUI

Esa configuración inicial (rutas y parámetros) es recogida por el módulo central *(MainWindow)* de la aplicación.

- Invoca al cargador de datos para leer geometrías y construye el mapeo VIS↔XL. Este mapeo es necesario porque la DLL y la aplicación trabajan con geometrías de distintos tamaños, una detallada para la simulación en la DLL (XL) y otra optimizada para la visualización interactiva (VIS).
- A continuación, el gestor de renderizado (*RenderManager*) recibe la geometría seleccionada y crea el mallado inicial que se muestra en la Visualización 3D.

1. **Generación de archivos mediante MATLAB**

   Si el usuario ha optado por generar curvas de APD (Action Potential Duration) o condiciones iniciales desde MATLAB, lanza el worker de generación MATLAB:

   - Este arranca el motor MATLAB, carga el modelo geométrico (CORGPUModel) desde el .mat, ejecuta los scripts y produce 'APD_curve.txt' e 'initial_conditions.txt', que contienen la información básica que el motor de simulación necesita para arrancar con unas condiciones fisiológicas coherentes y conociendo la respuesta dinámica del modelo cardíaco.
   - Una vez listos, devuelve las rutas a estos ficheros para incluirlos en la simulación.

2. **Creación y arranque de la simulación en la DLL**

   MainWindow pasa a su vez a la lógica de simulación:

   - Las rutas de los archivos generados, junto con el directorio de salida.

- Inicia la creación de la simulación en la DLL, obtiene el identificador de la simulación creada (como un "nombre" específico que usa la DLL para referirse a cada simulación que se crea) y el número de partículas, que informa del número de nodos.
- Solicita un primer frame de datos XL a la DLL y lo guarda en un repositorio.

3. **Bucle de simulación y buffer**

   MainWindow lanza en paralelo un hilo que:

   - Llama periódicamente a la DLL para obtener datos nuevos.
   - El hilo principal, a través de un temporizador GUI, consume esos datos y mantiene un buffer de índices de frame.

4. **Renderizado 3D**

   Cada vez que entra un nuevo índice en el buffer, *MainWindow*:

   - Recupera el frame de la geometría XL y mapea a la geometría VIS de la aplicación mediante un archivo de correspondencia de geometrías de tipo .mat.
   - Llama a *RenderManager* que aplica el coloreado y muestra el estado actual en la vista principal.

5. **Flujo de ablación**

   En cualquier momento el usuario puede:

   - Dibujar ablaciones con la herramienta de dibujo, que actualiza una capa de preparación específica para el dibujo interactivo.
   - Aplicar máscaras externas (archivos .mat o .txt) desde el panel de configuración, acumulándolas en una capa de preparación para máscaras externas. Cuando se confirma, combina las máscaras visibles: capa de dibujo, capa de máscaras externas y la ablación ya existente en la simulación; las mapea a índices XL. La DLL actualiza su estado interno y, el gestor de renderizado refleja la ablación en la escena 3D.

6. **Cálculo de mapas funcionales**

   Si el usuario solicita mapas, *MainWindow* extrae un segmento de fotogramas crudos, lo mapea a VIS y lanza el worker de análisis MATLAB:

   - El worker transfiere los datos VIS a MATLAB, ejecuta el script de análisis y devuelve los mapas funcionales junto con sus colormaps.
   - El módulo central muestra estas mini-vistas en el panel auxiliar y permite “enviar” cualquiera de ellas a la vista principal para superponerla al modelo 3D.

7. **Gestión de escenarios de ablación**

   El usuario puede guardar el estado completo de la simulación como un “escenario”. Después, en el panel de previsualización puede reproducirlo y comparar múltiples escenarios en pantalla dividida, sin interrumpir la simulación original.

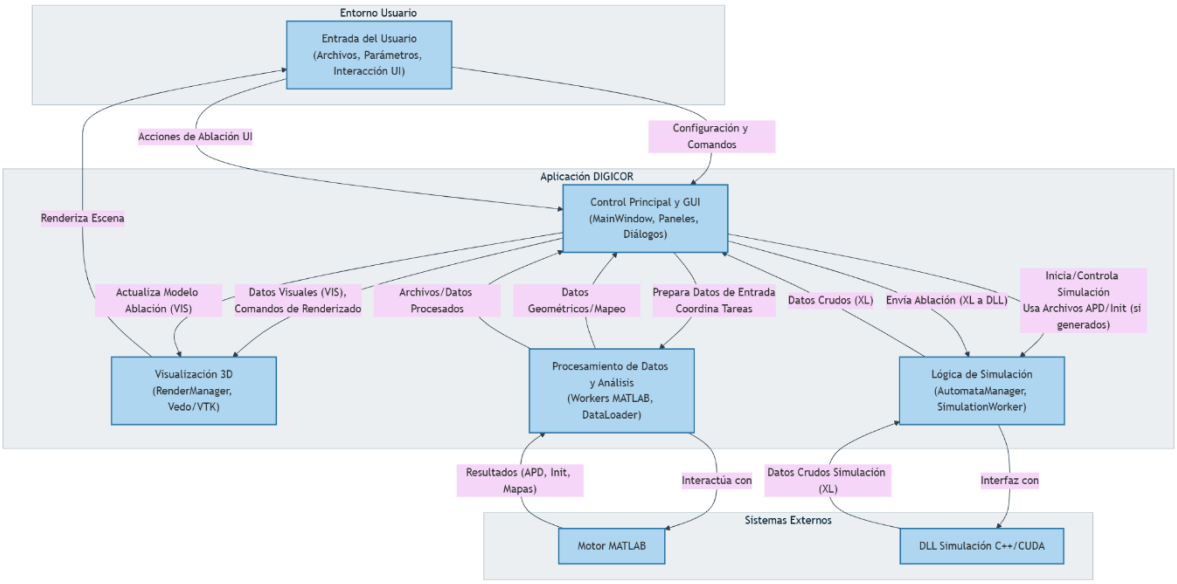


Figura 3.2 Flujo de datos de DIGICOR

Este flujo de datos, estructurado como diagrama y listado, aporta una visión clara de la secuencia metodológica en DIGICOR. Facilita la depuración y el mantenimiento al permitir localizar rápidamente cualquier punto de fallo o cuellos de botella en el procesamiento, y asegura que la interacción de usuario, simulación y visualización permanezcan completamente alineadas.

3.4. Herramientas de desarrollo

DIGICOR se ha desarrollado principalmente en Python, que actúa como entorno de orquestación entre los distintos componentes. Las tecnologías empleadas incluyen:

Python: núcleo lógico de la aplicación y sistema de gestión de eventos.

Python ha sido utilizado como núcleo lógico del sistema debido a su facilidad de uso, sintaxis clara y gran ecosistema de bibliotecas científicas y de visualización [40]. Su capacidad para integrar código compilado, como bibliotecas en C++ o módulos de MATLAB, ha permitido desarrollar una arquitectura flexible y eficiente. Python orquesta los eventos, gestiona la comunicación entre los módulos y permite una rápida iteración en el desarrollo.

PyQt5: construcción de la interfaz gráfica y herramientas de interacción.

Esta herramienta permite crear interfaces visuales robustas y multiplataforma, con widgets avanzados, herramientas visuales como Qt Designer y un sistema de eventos basado en señales y slots, ideal para el manejo modular de la lógica de interacción [41]. Además, su compatibilidad con contextos gráficos como VTK lo hace especialmente útil para integrar visualizaciones 3D en tiempo real dentro de la aplicación.

Frente a otras alternativas como Tkinter o wxPython, PyQt5 proporciona una mayor sofisticación visual, mayor personalización de componentes y mejor integración con bibliotecas de visualización avanzadas. Esto ha resultado en una interfaz más intuitiva, profesional y con un rendimiento estable.

VTK + Vedo: renderizado 3D, interacción con la geometría y colorimetría en tiempo real.

DIGICOR incluye un motor de visualización basado en **VTK (Visualization Toolkit)**, una biblioteca estándar en visualización médica y científica, y su envoltorio de alto nivel **Vedo**, que simplifica su uso desde Python [42][43]. Esta combinación permite el renderizado eficiente de geometrías cardíacas, la visualización de señales y mapas funcionales, así como la interacción fluida con modelos tridimensionales en tiempo real.

Se eligieron estas bibliotecas debido a su alto rendimiento (al estar escritas en C++), su capacidad para representar estructuras complejas y su compatibilidad con PyQt5. Frente a otras soluciones más simples o centradas en 2D, como Matplotlib, VTK y Vedo ofrecen un entorno más adecuado para la visualización biomédica avanzada.

C++ / CUDA (DLL): simulación eficiente mediante autómata celular.

El motor de simulación se ha implementado en **C++** para garantizar un alto rendimiento, y se ha acelerado mediante **CUDA**, la plataforma de computación en GPU de NVIDIA [44][45]. Esta combinación permite simular en tiempo real la propagación de la actividad eléctrica en una malla tridimensional compleja mediante un modelo de autómata celular.

El uso de C++/CUDA se justifica por su capacidad para ejecutar millones de operaciones simultáneamente, aprovechar la arquitectura paralela de las GPU y permitir un control fino sobre la memoria y los recursos computacionales. Además, encapsular este motor como una **biblioteca dinámica (DLL)** facilita su reutilización y modularidad, ya que puede ser invocado directamente desde Python mediante interfaces estandarizadas.

Esta decisión ha sido clave para lograr simulaciones interactivas sin comprometer el rendimiento, algo que no sería posible usando únicamente lenguajes interpretados.

MATLAB, **NumPy / SciPy / h5py**: generación de condiciones iniciales y mapas funcionales, gestión de arrays, búsqueda espacial, lectura/escritura de archivos .mat.

DIGICOR emplea tanto **MATLAB** como bibliotecas científicas de Python (**NumPy**, **SciPy**, **h5py**) para generar y analizar mapas funcionales derivados de la simulación. En este caso, el uso de MATLAB responde a la necesidad de integrar código ya existente desarrollado previamente por el grupo COR y Corify, el cual implementa funciones específicas de análisis y visualización en el ámbito de la simulación cardíaca. Esta decisión permitió reutilizar herramientas validadas y acelerar el desarrollo de la plataforma, asegurando compatibilidad con el flujo de trabajo actual.

Las bibliotecas de Python complementan esta funcionalidad y permiten desarrollar una versión más flexible y abierta del análisis, evitando la dependencia exclusiva de software propietario. **NumPy** ofrece estructuras de arrays eficientes y operaciones vectorizadas [46], **SciPy** proporciona algoritmos avanzados para análisis numérico y procesamiento de señales [47], y **h5py** permite guardar y compartir grandes volúmenes de datos en formato HDF5, asegurando compatibilidad con MATLAB y otros entornos. Estas herramientas garantizan la eficiencia en la manipulación de datos simulados, así como su análisis posterior.

En conjunto, la elección y combinación de estas tecnologías responde a una estrategia de equilibrio entre rendimiento, flexibilidad, compatibilidad e integralidad, elementos esenciales para una herramienta como DIGICOR destinada a la simulación biomédica interactiva en tiempo real.

3.5. Diseño de la interfaz de usuario

La interfaz gráfica de DIGICOR se concibió para ser intuitiva y accesible tanto a investigadores como a clínicos con escasa experiencia en simulación. Se centró en seis ejes metodológicos:

1. **Arquitectura de paneles y modos de visualización**

   Se eligió PyQt5 con paneles acoplables para que el usuario reorganice libremente su espacio de trabajo. El área central (Figura 3.3(1)) presenta dos modos: "simulación única" y "comparación de escenarios", de manera que alternar entre ambos resulta tan sencillo como pulsar un botón.

2. **Carga y gestión de geometrías**

   A través de un diálogo modal, el usuario puede elegir entre múltiples mallas contenidas en un solo archivo .mat. Internamente, un componente especializado extrae todas las estructuras disponibles y prepara los mapeos necesarios para la visualización y la simulación de forma transparente. Las geometrías se mostrarán una vez cargadas en el *RightControlPanel* **(**Figura 3.3(3)**)**.

3. **Control de simulación y reproducción**

   La simulación se ejecuta en un hilo independiente que alimenta un buffer de fotogramas. Un temporizador en la GUI consume ese buffer y actualiza la vista 3D. Los controles para esta funcionalidad («play/pause», avance por cuadro, ajuste de velocidad) se encuentran en el *BottomControlPanel* (Figura 3.3(2)), garantizando una experiencia fluida sin bloquear la interfaz.

4. **Interacción de ablación**

   Se desarrolló una herramienta de dibujo sobre el modelo 3D con pincel de tamaño ajustable. Los trazos se gestionan en varias capas: efectiva, staging externo, staging interactivo y backup. La capa efectiva refleja el tejido ablacionado en la DLL, la de staging externo acumula el efecto de las máscaras predefinidas, pero no aplicadas todavía y la capa de staging interactivo almacena los trazos hechos por el usuario que aún no se han enviado a la DLL. La capa de backup guarda el estado de dibujo al iniciar la edición, permitiendo deshacer los cambios efectuados.

5. **Visualización y comparación de escenarios**

   Para evaluar estrategias de ablación, el usuario puede guardar el estado completo de la simulación y reproducirlo en pestañas dedicadas (Figura 3.3(4)). Además, el modo de comparación principal muestra múltiples escenarios simultáneamente, facilitando la identificación visual de la estrategia más efectiva.

6. **Retroalimentación contextual y ayuda**

   Se incorporaron tooltips detallados en todos los controles, una barra de estado que informa en tiempo real de la acción en curso y un diálogo de ayuda rápida (accesible con F1) que resume atajos de teclado y flujo de trabajo. Para ver los atajos de teclado, véase el Anexo A.

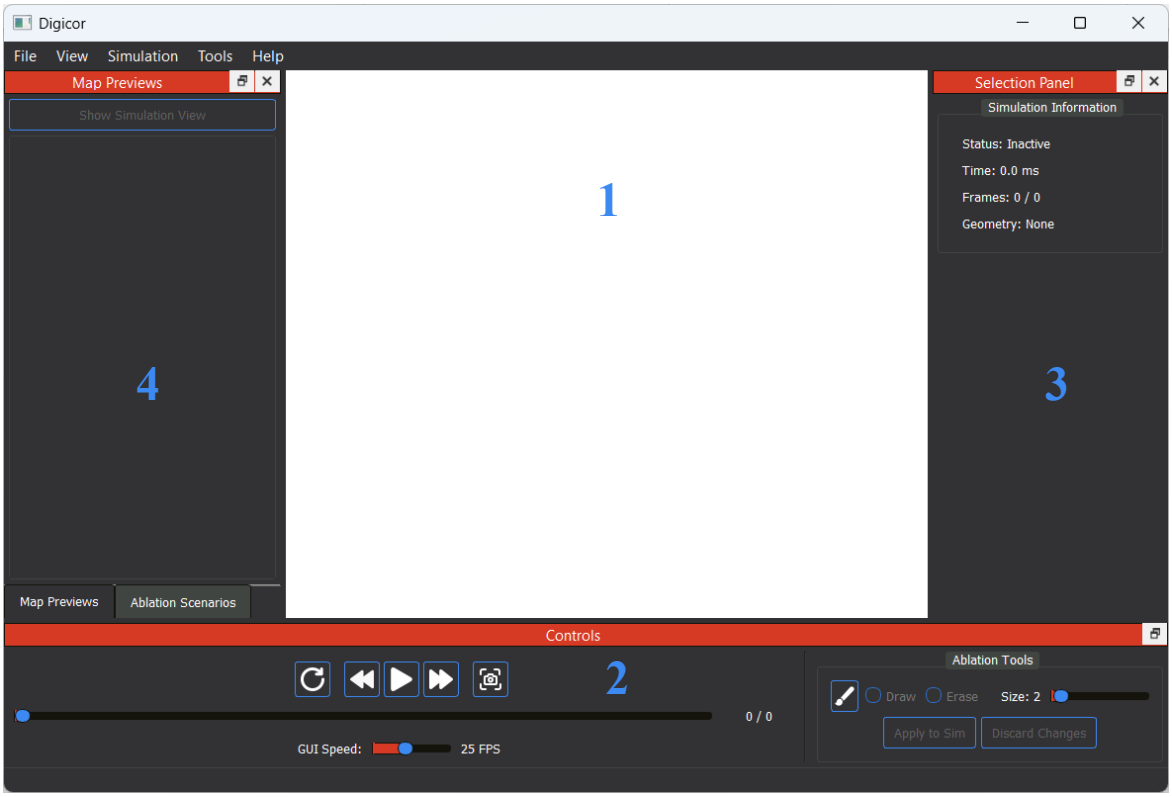


Figura 3.3 Interfaz Gráfica de Usuario (GUI) principal de DIGICOR. (1) Panel central de visualización 3D. (2) Panel de controles inferior (BottomControlPanel. (3) Panel de selección lateral derecho (RightControlPanel). (4) Paneles de previsualización laterales izquierdos (Docks con Pestañas).

Con esta aproximación metodológica se garantiza que la interfaz no solo sea potente y flexible, sino también clara y fácil de usar en entornos clínicos.

3.6. Motor de simulación y gestión de datos

Para la construcción del motor de simulación de DIGICOR y la organización de los datos generados se siguió un diseño orientado a rendimiento y flexibilidad, manteniendo siempre la interactividad de la GUI.

1. Dualidad de representaciones geométricas

Para conciliar precisión y velocidad, se emplean dos mallas auriculares complementarias:

- **Malla XL** (alta resolución): tamaño = 47 812 nodos, definido por geometry.txt, y utilizada internamente por la DLL para modelar la propagación con detalle.
- **Malla VIS** (≈ 4 000 nodos): versión reducida para renderizado en Vedo/VTK en la interfaz.

La correspondencia entre ambas se gestiona mediante un archivo .mat precalculado con los índices que permite transferir los datos de XL a VIS, y viceversa, sin pérdida de trazabilidad. Además, sobre los vértices VIS se construye un KD-Tree (K dimensional tree) que actúa como un índice espacial optimizado de los vértices de la malla visual. Cuando el usuario "pinta" sobre la malla, en lugar de buscar linealmente entre miles de vértices para ver cuáles están bajo el pincel, el KD-Tree

permite localizar de forma muy rápida y eficiente los vértices exactos que están en la proximidad del punto de interacción, según el tamaño del pincel.

2. Núcleo de simulación en GPU

El núcleo de la simulación se ejecuta en una biblioteca compilada para GPU, encargada de avanzar el modelo de autómata celular. Se comunica con la aplicación principal mediante una interfaz genérica que abstrae funciones de inicio, avance de pasos y recuperación de datos

3. Almacenamiento y buffer de animación

Los datos XL del potencial de acción se almacenan secuencialmente, lo que facilita la navegación completa en el tiempo. Cuando se solicita mostrar un frame, se mapea XL→VIS bajo demanda, minimizando el uso de memoria.

4. Inicialización y parámetros de simulación

La configuración de la simulación recibe:

- **Archivos de entrada**:

  inputs.txt: Este archivo contiene parámetros de control para la propia DLL de simulación, como la duración total de la simulación, la frecuencia de guardado de datos, y configuraciones específicas del motor del autómata celular.

  geometry.txt: Define la topología de la malla de alta resolución (XL) sobre la cual se ejecuta la simulación. Especifica el número total de nodos y, la conectividad entre ellos, lo cual es esencial para propagación del autómata celular y para la expansión de lesiones de ablación.

  APD_curve.txt: Proporciona la forma de la curva de duración del potencial de acción base. Este archivo define cómo varía el potencial transmembrana a lo largo del tiempo durante un ciclo de excitación-repolarización.

  initial_conditions.txt: Especifica el estado inicial de cada nodo en la malla XL al comienzo de la simulación. Si no se proporciona, la DLL puede usar un estado de reposo por defecto o una inicialización basada en otros parámetros.

- **Parámetros fisiológicos**:

  Coeficiente de difusión: representa la velocidad de conducción del impulso eléctrico a través del tejido auricular. Un valor más alto implica una propagación más rápida del frente de onda, mientras que un valor más bajo simula una conducción enlentecida.

  Frecuencia dominante basal: influye en la tasa intrínseca de activación o la excitabilidad de las células auriculares, relacionándose con la duración del potencial de acción y el período refractario.

  Desviación estándar de activación: introduce variabilidad o irregularidad en los tiempos de repolarización de las células. Una mayor dispersión de la refractariedad incrementa la probabilidad de bloqueo de conducción unidireccional y la fragmentación de los frentes de onda, factores que promueven la iniciación y el mantenimiento de la FA.

5. Aplicación dinámica de ablaciones

Cada vez que el usuario confirma una ablación:

1. Se combinan las capas de máscara VIS, interactiva, externa y efectiva, y se mapean a índices XL, incluyendo una expansión de vecindad. El objetivo de la expansión es "engrosar" la lesión en el dominio XL, asegurando que la región ablacionada en la malla de alta resolución sea proporcionalmente representativa de la malla VIS y forme una barrera de conducción más robusta y realista.
2. *RenderManager* envía los nodos seleccionados a la DLL, que ajusta el estado interno de esos elementos.
3. La GUI refleja inmediatamente el cambio, manteniendo la coherencia entre simulación y visualización.

3.7. Evaluación de estrategias de ablación

La evaluación de estrategias de ablación en DIGICOR se planteó en base a dos pilares complementarios: la observación interactiva de la simulación y el análisis de mapas funcionales. Con ellos, el usuario puede comparar distintas configuraciones de ablaciones antes de su aplicación clínica.

Simulación y comparación interactiva

Tras aplicar una o varias lesiones al modelo 3D, el usuario recorre la simulación en el tiempo para observar cómo cambian los patrones de activación. Para profundizar en el contraste entre diferentes abordajes, DIGICOR permite guardar “escenarios de ablación” que son copias de simulaciones. Cada escenario puede revisarse por separado en un panel de previsualización o visualizarse en paralelo en la vista principal, gracias a un modo multi-panel. Esto facilita la identificación visual de la estrategia que mejor suprime la actividad fibrilatoria, detiene rotores o reorganiza la dinámica eléctrica.

Generación de mapas funcionales

Para un análisis más objetivo, el usuario selecciona un intervalo temporal de la simulación y extrae los datos de potencial de acción crudo. Estos datos se transforman al dominio de visualización y se envían a MATLAB, donde se ejecutan scripts validados del grupo ITACA-COR. El resultado son tres mapas principales:

- **Frecuencia dominante (Mean DF)**: identifica en cada nodo la frecuencia más repetitiva de activación durante el intervalo, destacando focos de actividad rápida.
- **Mapa de estabilidad de la FA (Stability Map)**: señala regiones con actividad de alta frecuencia sostenida que pueden actuar como mantenedores de la fibrilación.
- **Histograma de patrones reentrantes (Reentrant Pattern Histogram)**: cuantifica la aparición de patrones rotacionales o de reentrada en cada zona, ayudando a localizar mecanismos.

Cada mapa se muestra primero en miniaturas dentro de su propio renderizador (Figura 3.4). El usuario puede luego aplicar cualquiera de ellos a la vista principal 3D.

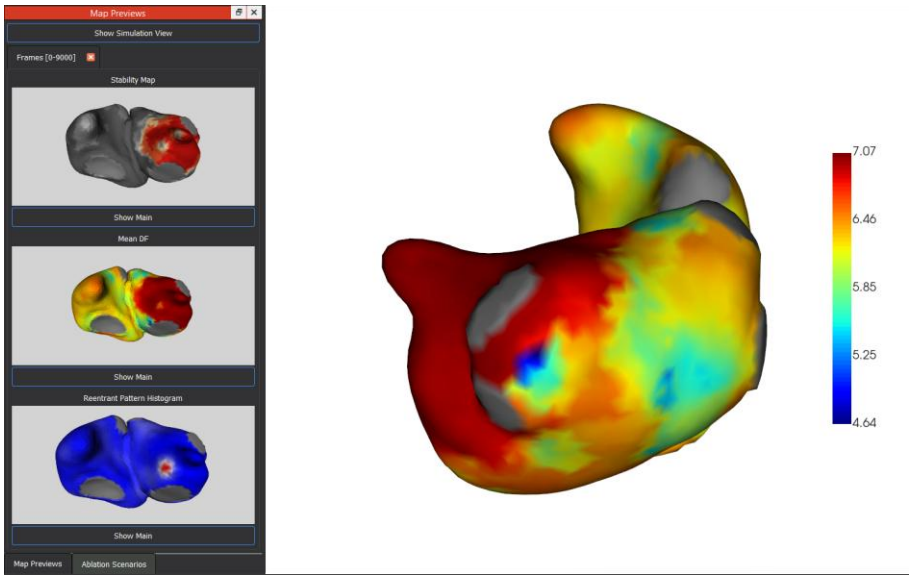


Figura 3.4 Visualización de mapas funcionales en DIGICOR. A la izquierda (panel "Map Previews"), se muestran miniaturas interactivas para cada mapa calculado. El mapa Mean DF se está visualizando en la vista principal. El botón "Show Simulation View" permite volver a la simulación dinámica.

Ciclo de feedback visual

En todo momento, la aplicación actualiza instantáneamente la representación de ablaciones y mapas, proporcionando un ciclo de retroalimentación rápido entre acción (definición de lesiones, elección de intervalo, comparación de escenarios) y resultado visual. Este enfoque interactivo sustituye, en primera instancia, la necesidad de análisis cuantitativos automatizados, dejando al experto la capacidad de valorar cualitativamente los efectos y decidir la estrategia óptima.

Con esta metodología, DIGICOR ofrece un entorno integral para diseñar, contrastar y perfeccionar estrategias de ablación antes de su validación clínica, combinando simulación en tiempo real, comparación paralela y herramientas avanzadas de visualización funcional.

3.8. Validación técnica y pruebas

La validación técnica de DIGICOR se centró en garantizar la correcta integración de sus componentes, la coherencia del procesamiento de datos y la estabilidad de la aplicación. Se diferenció entre pruebas de funcionalidad, validación cualitativa de simulaciones y estrés de la interfaz, dejando la validación clínica para estudios posteriores.

Pruebas de funcionalidad e integración

- **Recorrido de extremo a extremo**: Validación del flujo completo, desde la carga de archivos y parámetros, generación opcional de datos con MATLAB, ejecución en la DLL, mapeo XL→VIS, hasta la visualización de resultados y mapas.
- **Componentes GUI**: Comprobación de diálogos, controles de simulación, herramientas de dibujo y paneles de escenarios/mapas.

Validación cualitativa de la simulación

- **Escenarios de referencia**: Simulaciones con lesiones estándar para verificar bloqueos de conducción según lo previsto.
- **Detección de rotores**: Configuración de casos con actividad rotacional y contraste con el mapa de "Histograma de patrones reentrantes" para confirmar la correcta identificación de rotaciones.
- **Comparación bibliográfica**: Las salidas se cotejan con ejemplos de la literatura en fibrilación auricular.

Estabilidad y robustez

- **Pruebas prolongadas**: Animación continua durante largos períodos de tiempo para descartar fugas de memoria o degradación de rendimiento.
- **Dinámica de estado**: Cambio reiterado de geometrías, escenarios y máscaras para asegurar la consistencia interna sin bloqueos ni pérdidas de datos.
- **Casos límite**: Valores extremos de parámetros y rutas de excepción en la GUI para identificar y corregir fallos.

4. Desarrollo

4.1 Arquitectura interna de DIGICOR

La arquitectura de DIGICOR implementa la adaptación del patrón MVC descrito en la metodología, separando la lógica de simulación, la representación visual y la interacción del usuario a través de las siguientes clases y módulos principales:

En el **Modelo** implementado, la clase *AutomataManager* actúa como la pasarela directa con la biblioteca C++/CUDA (AutomataCUDAv1.dll), siendo responsable de la inicialización de la simulación (*create_simulation*), el avance de los pasos del autómata celular (*run_steps*), y la recuperación de los datos eléctricos (*get_data*). Los datos crudos de potencial de acción (formato XL), junto con los parámetros fisiológicos definidos por el usuario y las diversas capas de representación de la ablación, se almacenan secuencialmente en la estructura de datos *raw_history_storage*. Esta implementación permite que cualquier instante de la simulación sea accesible para análisis o reproducción posterior. La lógica de ejecución de la simulación en segundo plano es gestionada por *SimulationWorker*, mientras que la interacción con MATLAB la manejan *MatlabGenerationWorker* y *MatlabAnalysisWorker*, generando los datos y haciendo el análisis funcional, respectivamente.

La **Vista** se materializa a través de la clase *MainWindow* (que actúa como contenedor principal de la GUI PyQt5) y la clase *RenderManager*. *MainWindow* integra los paneles acoplables que alojan los controles de usuario (*RightControlPanel*, *BottomControlPanel*) y las mini-vistas de previsualización (*MapPreviewWidget*, *ScenarioPreviewWidget*). La escena 3D central es gestionada por *RenderManager*, que utiliza para renderizar la geometría cardíaca, actualizar dinámicamente el color de los vértices según los datos de potencial de acción (vía *update_vertex_colors*), y superponer elementos informativos como texto y barras de escala.

El rol del **Controlador** es asumido principalmente por *MainWindow*. Esta clase centraliza la lógica de interacción, respondiendo a los eventos generados por los elementos de la interfaz gráfica (botones, sliders, menús) y a las interacciones 3D capturadas (*MyInteractorStyle)*. *MainWindow* traduce estas entradas del usuario en acciones sobre el Modelo o en peticiones de actualización a la Vista. Para mantener una respuesta adecuada en la interfaz, la comunicación con los *workers* que ejecutan tareas se realiza de forma asíncrona, garantizando una experiencia de usuario fluida.

Para una representación visual detallada de la interconexión entre las clases y módulos específicos que componen esta arquitectura, véase el Diagrama de arquitectura del software en la Figura 4.1.

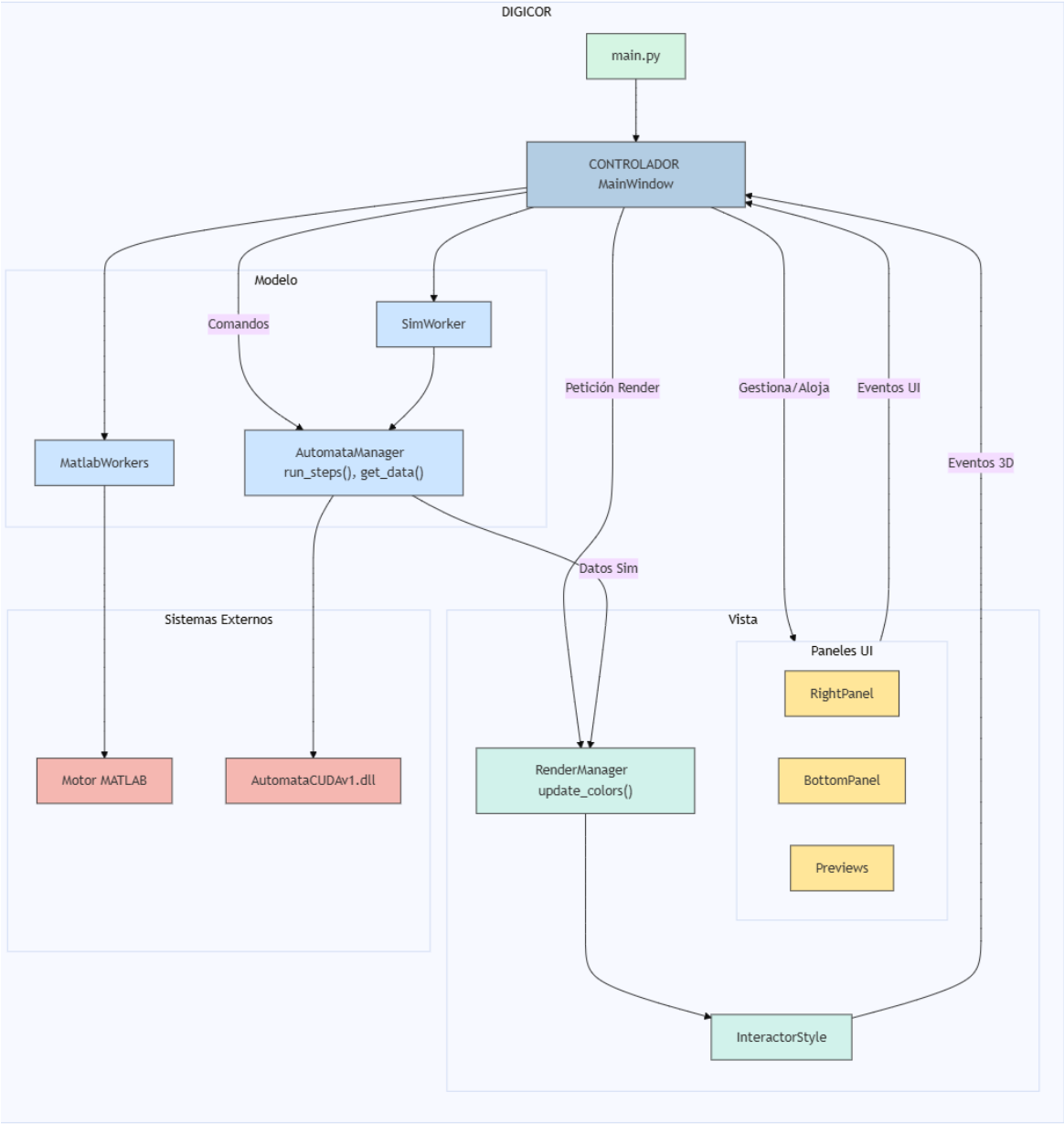


Figura 4.1 Diagrama de arquitectura del software

4.2 Frontend e interfaz de usuario

La interfaz gráfica de DIGICOR ha sido construida sobre el framework PyQt5, integrando múltiples elementos de control y visualización orientados a ofrecer una experiencia interactiva intuitiva y especializada para el estudio de la fibrilación auricular. La ventana principal (*MainWindow*) articula la estructura del frontend, agrupa menús, barra de estado y un conjunto de paneles reubicables donde el usuario organiza su espacio de trabajo.

4.2.1 Ventana principal y paneles

MainWindow organiza su contenido visual y de control mediante un sistema de paneles acoplables y un área central dedicada a la visualización 3D. Todos los paneles son reubicables, lo que permite adaptarlos al gusto del usuario, así como, ocultar o mostrar según la necesidad.

- **Panel central**: Es el corazón de la visualización 3D. Tiene dos modos, uno se asocia a *plotter_single* para la visualización de la simulación individual mapas estáticos o escenarios individual (Figura 4.2.a) y otro se asocia a *plotter_compare* para el modo de comparación de múltiples escenarios de ablación (Figura 4.2.b), donde el plotter se configura con múltiples viewports.

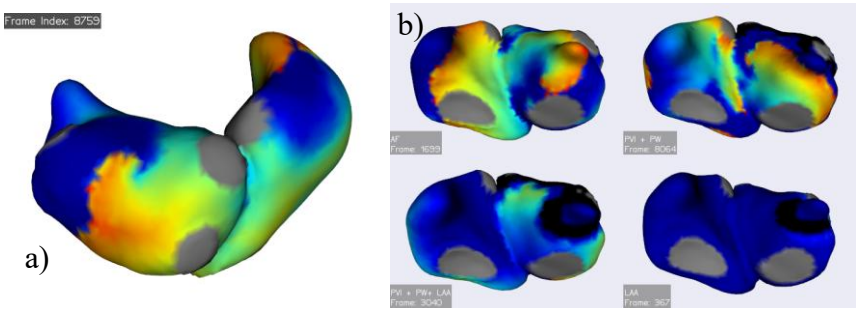


Figura 4.2 (a) Panel central en modo de visualización individual. Ejemplo mostrando una simulación de FA. (b) Panel central en modo de comparación de escenarios. Ejemplo de la disposición multi-panel (2x2 viewports) donde se visualizan simultáneamente cuatro escenarios de ablación diferentes. Cada viewport incluye una etiqueta con el nombre del escenario y el frame que se está mostrando de ese escenario concreto.

- **Paneles de control y selección :**

El *RightControlPanel*, llamado 'Selection Panel' se aloja a la derecha (Figura 4.3). Este panel muestra (de arriba abajo): información en tiempo del estado de la simulación, la lista de geometrías visuales disponibles, los modos de aplicación para máscaras externas ("Replace Staging", "Add to Staging"), y la lista de máscaras de ablación externas precargadas con botones para previsualizarlas (icono de ojo) o aplicarlas a la preparación.

El *BottomControlPanel*, llamado 'Controls' está acoplado en la parte inferior de la interfaz (Figura 4.4). Contiene los controles de reproducción (reinicio, anterior, play/pausa, siguiente), el slider de navegación temporal (indicando frame actual/total), y el slider de ajuste de velocidad (FPS). Adicionalmente, se ha integrado un botón de captura de pantalla directamente en la vista 3D para exportar la visualización actual como una imagen de alta resolución. A la derecha, se ubica el grupo "Ablation Tools" con el botón de activación del modo edición (pincel), los selectores "Draw"/"Erase", el slider de tamaño del pincel, y los botones para aplicar o descartar los cambios.

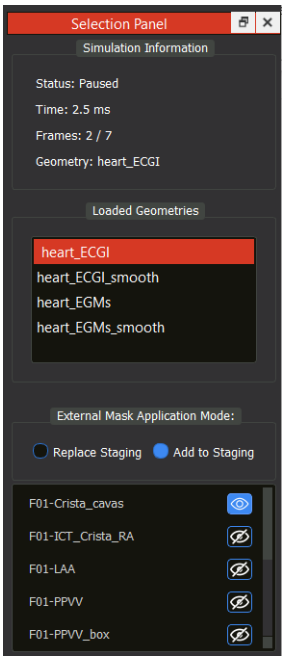


Figura 4.3 Panel de Selección Lateral Derecho (Selection Panel).

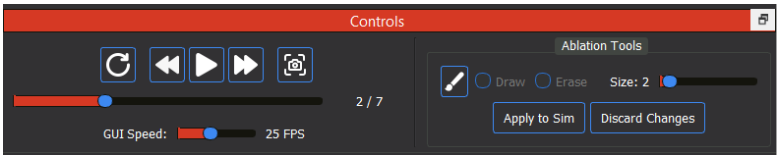


Figura 4.4 Panel de controles inferior (Controls). Parte izquierda para control de la animación e izquierda para control de la ablación

- **Paneles de previsualización y análisis detallado:**
  - Localizados típicamente en el lateral izquierdo y gestionados mediante pestañas tabuladas (Figura 4.5), estos paneles permiten una exploración más profunda de resultados específicos sin alterar la vista principal.
  - **Pestañas de mapas funcionales (MapTabWidget):** Cada pestaña, agrupa varios *MapPreviewWidget*. Cada *MapPreviewWidget* ofrece una vista 3D renderizada por Vedo del mapa funcional correspondiente para un evento de cálculo específico, con su propia barra de escala y la opción de mostrar dicho mapa en el panel central (Figura 4.5.a).
  - **Pestañas de escenarios de ablación (ScenarioTabWidget):** Cada pestaña contiene un *ScenarioPreviewWidget* que permite la reproducción y navegación independiente de un escenario de ablación previamente guardado (*ablation_scenarios*). Estos widgets también cuentan con su propio renderizador Vedo y controles, y pueden solicitar que su escenario se muestre en la vista principal o se añada al modo comparación (Figura 4.5.b).

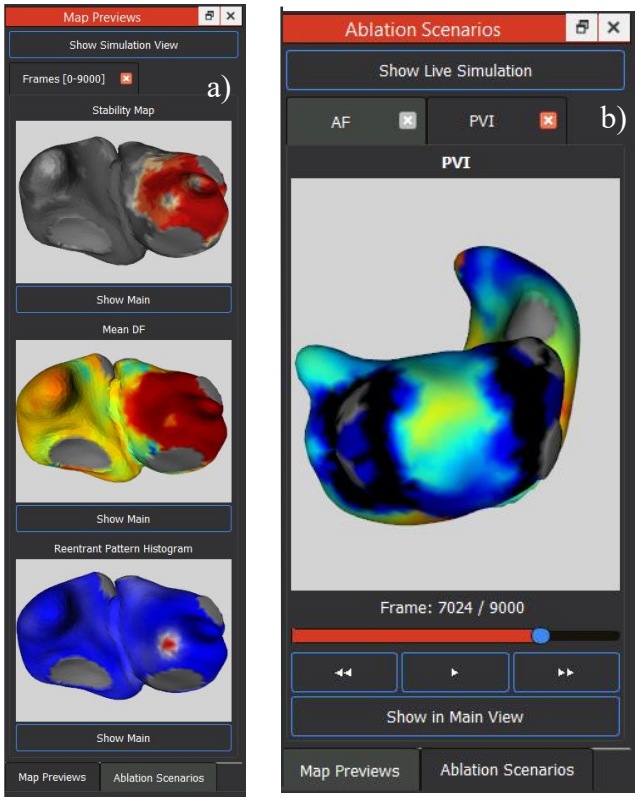


Figura 4.5 Panel de previsualización izquierdo de la interfaz, los dos modos están tabulados como se ve en la parte inferior. (a) Panel de Previsualización de Mapas Funcionales en su pestaña activa. (b) Panel de Previsualización de Escenarios de Ablación.

4.2.2 Visualización 3D e interacción

La representación 3D de las geometrías auriculares y los datos de simulación se realiza mediante objetos *Mesh* (malla) de Vedo, construidos a partir de los arrays de vértices y caras cargados desde los archivos .mat. Estos objetos son gestionados e integrados en la escena por la clase *RenderManager*.

1. **Renderizado dinámico (*RenderManager*):**
    - *setup_initial_scene()* y *setup_basic_scene()* se encargan de establecer la geometría base en el *Plotter* activo cuando se inicia una simulación o se carga una nueva geometría. La 'basic' es para visualización estructural o cuando no hay datos APD específicos del frame actual y la 'initial' es específicamente para cuando se tiene el primer frame de datos APD de una simulación.
    - *update_mesh_from_state()* es invocado continuamente para actualizar los colores de los nodos de la malla en función del potencial de acción y la máscara de ablación combinada. Esto se logra internamente mediante la función *update_vertex_colors*, que mapea los valores escalares a una paleta de colores (generalmente 'jet') y aplica un color distintivo (negro) a los nodos ablacionados.
    - El *RenderManager* también gestiona la visualización de mapas estáticos (*display_static_map_with_ablation*), la superposición de texto 2D informativo (como el índice del frame) y la alternancia del modo *wireframe*.
2. **Interacción con la malla 3D (*MyInteractorStyle*, *DrawingTool*):**
    - Para facilitar la definición interactiva de lesiones, se implementó un estilo de interacción VTK personalizado, *MyInteractorStyle*. Este estilo, asignado al *Plotter* principal, permite capturar y procesar eventos del ratón, específicamente el clic derecho y el arrastre, para la edición.
    - Cuando el modo de ablación interactiva está activo (*ablation_mode_active*), *MyInteractorStyle* delega estos eventos a la clase *DrawingTool*.
    - *DrawingTool* realiza una operación de *picking* sobre la malla que puede estar oculta o activa (Figura 4.6) para identificar el punto exacto de interacción del usuario en la superficie 3D y, utilizando el *KDTree* asociado a la geometría actual, selecciona los N vértices visuales más cercanos. Estos vértices se marcan o desmarcan en la capa de staging interactivo según si el usuario está en modo dibujo o borrado (*eraser_mode_active*). La retroalimentación visual de esta capa de *staging* es inmediata, ya que *MainWindow* es invocado para refrescar la escena 3D.

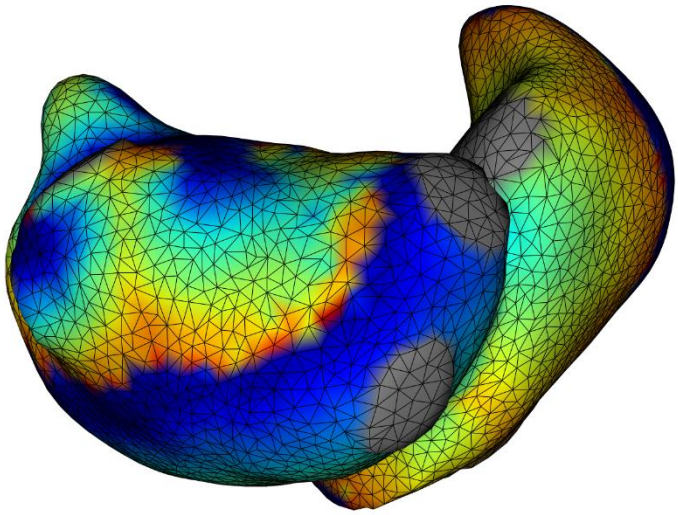


Figura 4.6 Visualización de la geometría con la malla activa (wireframe).

4.2.3 Reproducción y navegación temporal

La gestión de la reproducción y la navegación por el historial de la simulación es un componente central de la interfaz de DIGICOR, diseñado para proporcionar al usuario un control total y una experiencia visual fluida. Esta funcionalidad se articula a través de varios elementos y lógicas clave en el frontend:

- **Controles de Usuario (*BottomControlPanel*):** El usuario interactúa con la simulación a través de los widgets del panel de control inferior. Esto incluye:
    - **Botones de Reproducción:** Un botón de "Play/Pause" para iniciar o detener la animación.
    - **Navegación por Frames:** Botones de "Anterior" y "Siguiente" para un avance o retroceso manual, frame a frame.
    - **Navegación por Tiempo (*QSlider*):** Un slider de tiempo que permite al usuario saltar instantáneamente a cualquier punto del historial de la simulación que ya ha sido generado.
    - **Control de Velocidad (*QSlider*):** Un slider que ajusta los fotogramas por segundo (FPS) a los que se reproduce la animación, controlando la velocidad de visualización.
- **Temporizador de la GUI (*gui_update_timer*):** La reproducción animada está gobernada por un *QTimer* en el hilo principal. Cuando el usuario pulsa "Play", este temporizador se activa y comienza a emitir señales a una frecuencia determinada por el "GUI Speed slider". Es este temporizador el que marca el ritmo de la visualización, actuando como el "marcapasos" del frontend.
- **Actualización de Frames (*_update_gui_animation_frame*):** Este es el método (slot) conectado al *gui_update_timer*. En cada "tick" del temporizador, este método es responsable de

avanzar la simulación un paso. Para ello, determina el siguiente índice de frame a mostrar y solicita su visualización.

- **Renderizado de Frames *(_show_historical_frame*):** Esta función es la encargada final de la visualización. Recibe un índice de frame, recupera los datos crudos correspondientes del repositorio principal (*raw_history_storage*), los mapea a la geometría visual (VIS) y delega la tarea de coloreado y renderizado 3D a la clase *RenderManager*. También actualiza los indicadores en la GUI, como el número de frame actual en el slider y las etiquetas de tiempo.
- **Sincronización con el Backend**: El frontend está diseñado para estar completamente desacoplado del backend, asegurando una experiencia de usuario fluida e ininterrumpida. La interacción con el *SimulationWorker* se gestiona a través de una caché predictiva (*animation_buffer).* Mientras el usuario reproduce la animación, el método *update_gui_animation_frame* avanza por el historial (*raw_history_storage),* simultáneamente*,* un mecanismo de control *(_check_and_manage_worker)* monitoriza el estado de la caché en segundo plano. Si detecta que la cantidad de frames precargados en la caché desciende por debajo de un umbral, activa al *SimulationWorker* para que genere nuevos lotes de datos y rellene la caché. De esta forma, la caché se mantiene contantemente llena y cuando la reproducción de acerca al final del historial, se activa un mecanismo de transferencia que mueve los datos de la caché al historial.

Este diseño centrado en el frontend asegura que, desde la perspectiva del usuario, la interacción con el tiempo de la simulación sea siempre intuitiva, directa y fluida, ocultando la complejidad de la generación de datos asíncrona que ocurre en el backend.

4.2.4 Paneles auxiliares

Además de la vista 3D principal y los paneles de control primarios, la interfaz de DIGICOR se enriquece con widgets especializados que facilitan el análisis detallado y la comparación de resultados.

- **MapPreviewWidget**: Tras el cálculo de mapas funcionales, se instancia un *MapPreviewWidget* para cada mapa generado. Cada *MapPreviewWidget* contiene su propio renderizador Vedo, permitiendo al usuario explorar el mapa tridimensionalmente (zoom, rotación) de forma independiente, sin afectar la vista central. Incluye una barra de escala ajustada a los datos del mapa y un botón para solicitar que dicho mapa se muestre en la vista 3D principal para una inspección más detallada.
- **ScenarioPreviewWidget**: Cuando un usuario guarda un estado de simulación como un escenario de ablación (*ablation_scenarios*), o al cargar escenarios previamente guardados, se crea un *ScenarioPreviewWidget*. Este widget permite la reproducción, navegación temporal y control de la visualización del escenario de forma aislada, con sus propios controles y renderizador Vedo. Además, emite señales que permiten sincronizar su estado con la vista 3D principal o con el modo de comparación multi-panel. Estos widgets se crean y destruyen dinámicamente según acciones.

Visualización detallada de mapas funcionales en la interfaz

La representación de los mapas funcionales en DIGICOR se apoya en colormaps personalizados y herramientas gráficas diseñadas para representar con claridad los resultados sobre la malla 3D. Cada

tipo de mapa se renderiza con un esquema de color específico como se puede apreciar en la Figura 4.5 (a), el mapa de color puede ser importado desde archivos .mat desarrollados previamente o utilizar paletas estándar.

La aplicación del mapa funcional se realiza directamente sobre la geometría 3D auricular activa, asignando un color a cada vértice según su valor en el mapa, como por ejemplo la frecuencia dominante o un índice de actividad rotacional. Se da soporte al uso de colormaps específicos para ciertos análisis. Estos mapas de color pueden cargarse automáticamente desde archivos .mat durante el proceso de análisis llevado a cabo por *MatlabAnalysisWorker*. Para otros mapas, o como alternativa, se utilizan colormaps estándar (ej. 'jet', 'viridis') proporcionados por la biblioteca Vedo.

Gestión y comparación de estrategias de ablación mediante escenarios

Una funcionalidad destacada de DIGICOR, tanto desde la perspectiva metodológica como de la interfaz de usuario, es la capacidad de guardar estados completos de la simulación como "escenarios de ablación". Estos escenarios encapsulan la geometría activa en ese momento, el mapeo de índices, el segmento relevante del historial de datos crudos de simulación y el historial de las lesiones de ablación aplicadas hasta el punto de guardado. Una vez que un escenario es guardado, la simulación principal puede ser reseteada (*menu_restart_simulation*) o modificada para aplicar una nueva estrategia de ablación, permitiendo así generar y evaluar múltiples alternativas terapéuticas dentro de una misma sesión de trabajo.

Posteriormente, estos escenarios almacenados ofrecen diversas formas de evaluación. Se pueden consultar de forma individual (Figura 4.5(b)), donde cada escenario se carga y reproduce de manera independiente en su propia pestaña de previsualización (*ScenarioPreviewWidget*). Esto permite un análisis detallado de su evolución temporal y del efecto específico de la estrategia de ablación contenida en él. Adicionalmente, los escenarios pueden ser comparados visualmente de forma directa en la vista principal mediante el modo de "comparación de escenarios" (Figura 4.2(b)). Este modo sustituye temporalmente la escena de simulación única por una disposición multi-panel, donde cada viewport muestra un escenario diferente. Esto permite al usuario analizar y contrastar simultáneamente el efecto relativo de cada intervención sobre la misma base geométrica e historial de datos iniciales.

Este enfoque de gestión y comparación de escenarios facilita la exploración sistemática de alternativas terapéuticas y refuerza el valor de DIGICOR como herramienta de apoyo en la toma de decisiones, permitiendo al usuario identificar de manera más efectiva las estrategias que podrían conducir a una supresión óptima de la actividad fibrilatoria o a la reorganización eléctrica deseada.

4.3 Backend y gestión de simulación

El backend de DIGICOR constituye el núcleo computacional y de gestión de datos de la plataforma, diseñado para ejecutar simulaciones eléctricas, coordinar tareas de análisis y mantener una comunicación eficiente y asíncrona con el frontend. Su arquitectura multihilo es fundamental para permitir cálculos intensivos, como la evolución temporal de la arritmia o el análisis de datos en MATLAB, sin comprometer la respuesta de la interfaz de usuario.

4.3.1. Interfaz con el motor de simulación

La interacción directa con el motor de simulación externo, la biblioteca AutomataCUDAv1.dll, se centraliza en la clase *AutomataManager* que integra las siguientes funcionalidades.

- **Carga dinámica y definición de funciones:** *AutomataManager* carga dinámicamente la DLL mediante la librería ctypes de Python (permite la interoperabilidad entre Python y C++) y define la interfaz de llamada para cada función interpretable por la DLL, especificando los tipos de datos exactos de sus argumentos y de su valor de retorno. Esta definición es fundamental para la correcta interoperabilidad.
- **Gestión del ciclo de vida de la simulación:** Es responsable de invocar create*_simulation* con las rutas a los archivos de entrada y el directorio de salida. Tras la creación, almacena el manejador de la instancia de simulación y consulta el número total de partículas del dominio simulado. También gestiona la destrucción (*destroy_simulation*) y el reseteo (*restart_simulation*) de la instancia en la DLL.
- **Ejecución de pasos y recuperación de datos:** Proporciona métodos para avanzar la simulación por lotes de pasos temporales (*run_steps*) y para recuperar arrays representativos del estado eléctrico del sistema (*get_data),* como el potencial de acción, velocidad de conducción, etc., en el formato XL.
- **Actualización de ablaciones y expansión:** Recibe los índices de los nodos a ablacionar (ya mapeados a XL y potencialmente expandidos) desde *MainWindow* y los transmite a la DLL mediante *update*_ablation.
- **Manejo de errores:** Interpreta los códigos de error devueltos por la DLL y los traduce a mensajes descriptivos.

4.3.2 Gestión del Tiempo

La aplicación Digicor opera con distintos dominios temporales que son cruciales para entender su funcionamiento y rendimiento: el **tiempo simulado**, el **tiempo real de cálculo**, y el **tiempo de visualización**.

- **Tiempo simulado** Representa la progresión del modelo electrofisiológico cardíaco. Este avanza en pasos discretos de alta resolución, donde cada paso interno ejecutado por la biblioteca DLL corresponde a un avance de 0.125 milisegundos (ms), lo que equivale a una frecuencia de muestreo de 8 kHz. Para optimizar la comunicación y el rendimiento, la interacción de Python con la DLL se realiza en lotes. La variable correspondiente, configurada en 10 pasos, instruye al *SimulationWorker* para que la DLL ejecute este número de pasos internos antes de que se recupere el estado. El estado de la simulación (datos crudos del potencial de acción en el dominio XL) obtenido al final de cada lote se almacena como un "frame de historial" en la estructura raw*_history*_storage. Consecuentemente, cada frame en este historial representa un avance de 10 pasos × 0.125 ms/paso = 1.25 ms en el tiempo simulado. La interfaz gráfica muestra el tiempo simulado acumulado correspondiente al frame activo, calculado como Índice_del_frame × 1.25 ms.
- **Tiempo real de cálculo:** Corresponde al tiempo de “reloj de pared” que transcurre para que el sistema procese un lote completo. Esto incluye la ejecución de los 10 pasos de la DLL, la transferencia de los datos resultantes a Python, y la gestión interna por el *SimulationWorker*. Este tiempo, se mide experimentalmente durante la ejecución. Es importante notar que esta métrica no se visualiza directamente en la interfaz gráfica de usuario, pero es fundamental para el análisis.

- **Tiempo de aplicación y control de visualización:** El parámetro "GUI Speed" determina la velocidad con la que la interfaz gráfica presenta los resultados de la simulación, es como ajustar la velocidad de reproducción de un vídeo. El usuario puede elegir ver la evolución de los datos simulados más rápida o lentamente en pantalla. Sin embargo, esta configuración de visualización es independiente de la velocidad a la que el motor de simulación (DLL) calcula esos datos; la velocidad de visualización no altera la eficiencia ni el tiempo real del cálculo subyacente. Si el "GUI Speed" es inferior a la tasa de producción del backend: la simulación se ejecutará a la velocidad solicitada por el usuario. Si el "GUI Speed" es superior a la tasa de producción del backend: la animación intentará reproducirse a la velocidad solicitada, pero estará limitada por la rapidez con la que el backend pueda suministrar nuevos frames.

4.3.3 Ejecución asíncrona de la simulación

Para mantener una interfaz de usuario fluida y evitar bloqueos durante los cálculos de simulación, la ejecución de los pasos de la DLL se delega a la clase *SimulationWorker*. Esta clase se mueve a un hilo independiente (*SimulationWorkerThread*) gestionado por *MainWindow*. El funcionamiento del *SimulationWorker* se basa en un procesamiento por lotes controlado por eventos, en lugar de un bucle de ejecución continua, para una mejor integración.

- **Procesamiento por Lotes y Controlado por Temporizador** El SimulationWorker utiliza un *QTimer* interno. Cuando está en modo de "reproducción continua", este temporizador se dispara a alta frecuencia, invocando el método *process_one_batch()*. Este método, a su vez, llama a *run_steps()* para ejecutar un lote de pasos de simulación en la DLL y luego a *get_data()* para recuperar los resultados. Este enfoque basado en eventos evita el consumo innecesario de CPU y permite que el hilo del worker responda a otras señales de forma más eficiente.
- **Comunicación mediante Señales:** Tras procesar cada lote, *SimulationWorker* emite una serie de señales asíncronas de PyQt5 que son capturadas por *MainWindow* en el hilo principal:
    - *data_ready:* Envía el array de datos crudos (formato XL) del potencial de acción. Este dato se almacena en la caché predictiva (animation_buffer) para su posterior uso.
    - *step_completed*: Notifica el paso de tiempo actual de la simulación en la DLL.
    - *batch_processed:* Indica que el procesamiento de un lote ha concluido. Es fundamental para la sincronización, especialmente en la navegación manual frame a frame.
    - *error:* Informa a *MainWindow* sobre cualquier error ocurrido durante la interacción con la DLL, permitiendo una gestión de errores centralizada.
- **Control de Ejecución Externo:** *MainWindow* actúa como el controlador principal y gobierna el comportamiento del *SimulationWorker* a través de métodos que modifican los flags internos del worker. La función clave es *set_continuous_run(),* que actúa como un interruptor. *MainWindow* utiliza este método para activar o pausar la generación continua de datos por parte del worker, basándose en la lógica de la caché predictiva (es decir, si la caché necesita ser rellenada o si ya está llena).

Este diseño garantiza una separación clara de responsabilidades, donde *MainWindow* orquesta el flujo general y la visualización, mientras que *SimulationWorker* se encarga de interactuar con la DLL de forma asíncrona. Se puede ver de forma visual en la Figura 4.7.

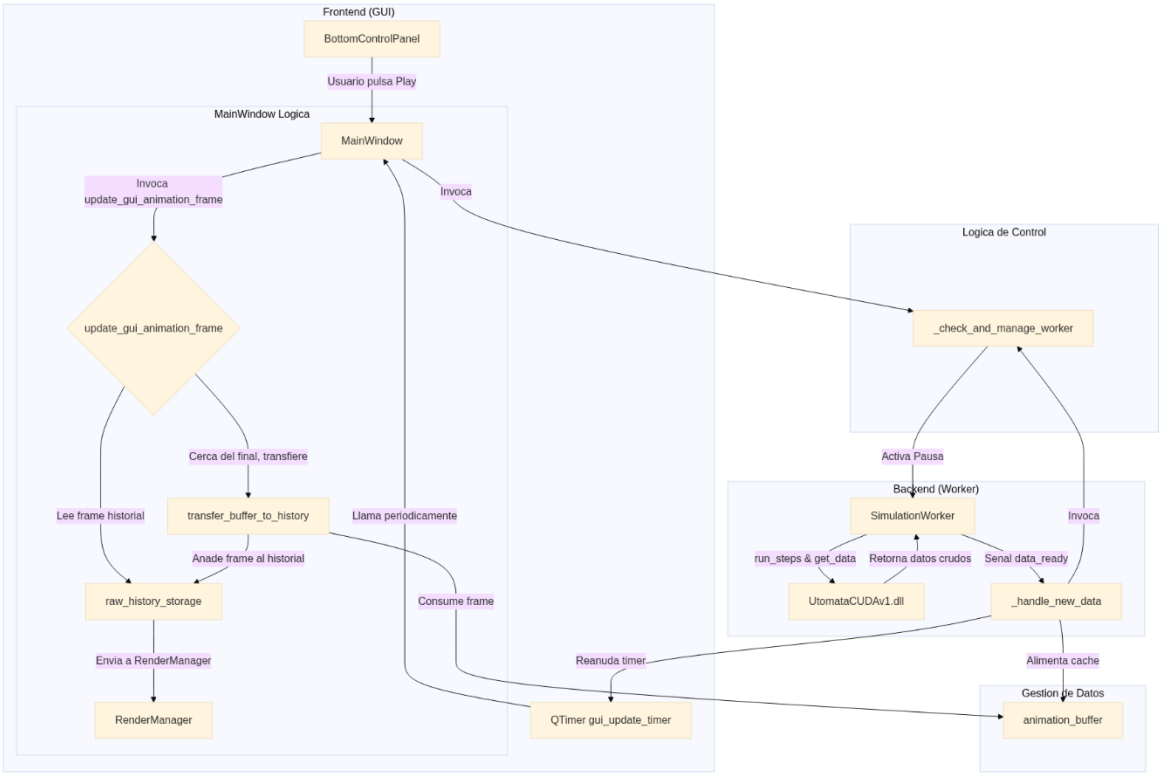


Figura 4.7 Diagrama de la estructura asíncrona de la simulación en DIGICOR.

4.3.4 Integración con MATLAB

DIGICOR integra funcionalidades que requieren el motor MATLAB para la generación de datos de entrada y el análisis post-simulación. Para asegurar que estas operaciones, potencialmente largas, no afecten la respuesta de la interfaz gráfica, se han implementado dos *workers* especializados, y ejecutándose en su propio hilo gestionado por *MainWindow*:

- **MatlabGenerationWorker:**
  - **Propósito:** Generar los archivos de entrada necesarios para la simulación**,** como la curva de duración del potencial de acción (APD_curve.txt) y las condiciones iniciales (initial_conditions.txt).
  - **Funcionamiento:** Recibe de *MainWindow* los parámetros definidos por el usuario en *ParameterDialog* y la ruta al script MATLAB correspondiente. Utiliza el motor *matlab.engine* para ejecutar dicho script en un entorno aislado. Los archivos resultantes se almacenan en un directorio temporal, y sus rutas se notifican a *MainWindow* mediante la señal *generation_finished*.
- **MatlabAnalysisWorker:**

- **Propósito:** Realizar análisis post-simulación sobre los datos generados, como el cálculo de mapas funcionales.
- **Funcionamiento:** Recibe de *MainWindow* los datos de simulación y la geometría visual correspondiente. Ejecuta scripts MATLAB de análisis, que devuelven los arrays de los mapas calculados y los colormaps personalizados. Estos resultados se emiten a *MainWindow* mediante la señal *finished*.

Ambos workers gestionan la comunicación con el motor MATLAB y emiten señales error en caso de problemas durante la ejecución de los scripts. Las señales *finished* y *error* también se utilizan para que *MainWindow* gestione adecuadamente el ciclo de vida de los hilos correspondientes.

4.3.5 Carga de datos, mapeo y utilidades

El backend de DIGICOR se apoya en módulos auxiliares especializados para la carga de datos, el mapeo entre dominios y otras operaciones comunes, centralizando estas funcionalidades para un uso eficiente y coherente a lo largo de la aplicación:

- **data_loader.py:** Este módulo contiene funciones especializadas para la carga y validación de datos desde archivos externos:
  - *find_geometries_in_mat():* Carga y valida múltiples estructuras geométricas (vértices y caras) y sus correspondientes indicadores de archivo desde archivos .mat.
  - *load_masks_from_mat_file()* y *load_masks_from_txt_folder():* Cargan y validan máscaras de ablación. Las máscaras .mat contienen índices visuales (VIS) de nodos no ablacionados, mientras que las máscaras .txt definen directamente los índices de simulación (XL) de los nodos a ablacionar. La validación se realiza sobre el número de vértices de la geometría visual o de la simulación activa, según corresponda.
  - *load_mapping_indices():* Carga y valida los vectores de mapeo de índices entre los dominios de simulación (XL) y visual (VIS) desde archivos .mat, cruciales para la correcta correspondencia de datos.
- **utils.py:** Este módulo proporciona funciones de utilidad general, destacando entre ellas:
  - *update_vertex_colors():* Aplica colorimetrías personalizadas o estándar a los vértices de una malla en función de datos escalares. Esta función es fundamental para la visualización, generando los arrays de colores RGBA que son utilizados por el *RenderManager* y los *Plotter* de los widgets de previsualización.

- Toda la gestión de los buffers de datos para la animación, el almacenamiento del historial completo de simulación, la sincronización de la animación, así como la lógica compleja para la aplicación, combinación y visualización de las diferentes capas de máscaras de ablación, se implementa en los componentes del backend (principalmente en *MainWindow* y las clases que esta orquesta, como *RenderManager* y *DrawingTool*). Esta centralización garantiza una separación efectiva entre la lógica de la aplicación y su presentación visual en el frontend

4.4 Flujo de trabajo interactivo

DIGICOR ha sido diseñado para guiar al usuario de forma intuitiva a través de un flujo de trabajo completo, desde la configuración inicial de una simulación hasta el análisis de resultados y la evaluación de estrategias de ablación post-intervención. La plataforma permite encadenar estas

operaciones de manera fluida con los módulos internos comunicándose asíncronamente para mantener la agilidad de respuesta de la interfaz en todo momento.

Una representación visual detallada de las principales secuencias de interacción del usuario con la aplicación, incluyendo la creación de simulaciones, la navegación temporal, la definición de ablaciones, el cálculo de mapas y la gestión de escenarios, se presenta en el Diagrama de flujo de experiencia de usuario en el Anexo [B].

4.4.1 Configuración inicial y carga de geometrías

La creación de una nueva simulación se inicia desde el menú "File", seleccionando "Create Simulation (Generate via MATLAB)" o "Create Simulation (from TXT Files)" como muestra la Figura 4.8. Este proceso guía al usuario a través de una secuencia de diálogos:

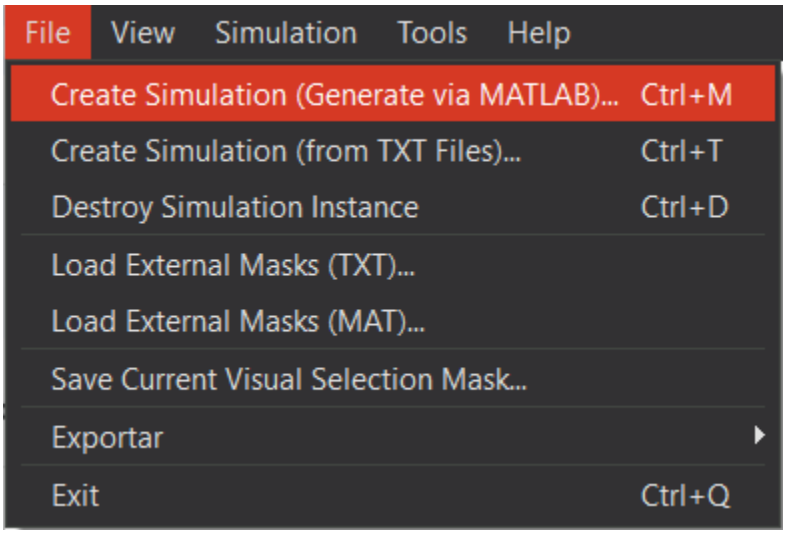


Figura 4.8 Barra de menú desplegada para mostrar cómo se puede crear una simulación.

1. Selección de archivos de entrada: El usuario selecciona los archivos necesarios para la simulación. Esto incluye los parámetros para el motor de cálculo (inputs.txt), la conectividad de la malla (geometry.txt), y un archivo .mat que contiene las geometrías visuales y sus mapeos. Dependiendo de la opción elegida, también seleccionará scripts .m de MATLAB para la generación automática de la curva APD y condiciones iniciales, o bien los archivos APD_curve.txt e initial_conditions.txt si ya existen.

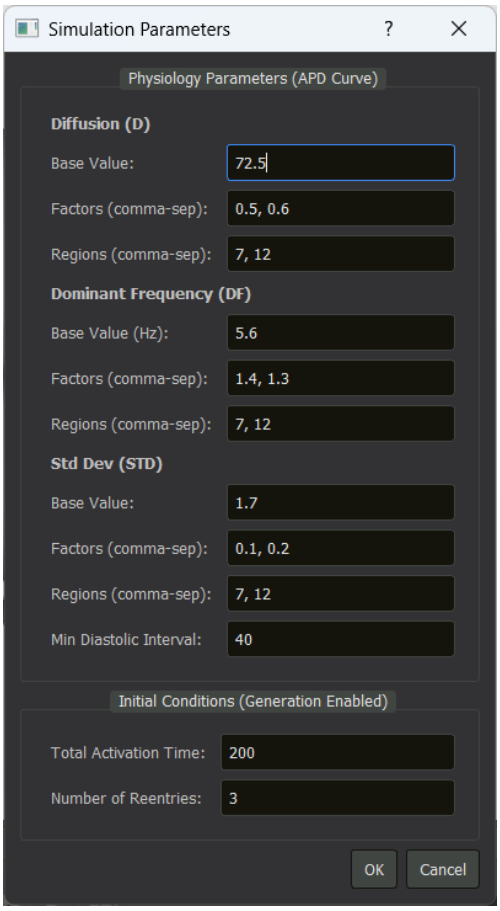


Figura 4.10 Diálogo de configuración de parámetros. Se pueden modificar los parámetros fisiológicos que generan el archivo APD_curve y los que generan las condiciones iniciales.

2. **Configuración de parámetros fisiológicos:** Se presenta el *ParameterDialog (*Figura 4.9), donde el usuario introduce o ajusta las constantes fisiológicas que definirán el comportamiento eléctrico del modelo. Se agrupan en secciones para el Coeficiente de Difusión (D), la Frecuencia Dominante (DF) y la Desviación Estándar del intervalo de activación (STD), permitiendo especificar valores base, factores de modificación y las regiones anatómicas donde estos se aplican. En la parte inferior, se configuran los parámetros para la generación opcional de condiciones iniciales mediante MATLAB.

3. **Selección de geometría visual:** A partir del archivo .mat proporcionado, se extraen todas las geometrías válidas (aquellas que tengan 4000 nodos conectados para formar caras triangulares). Se muestra el *GeometrySelectionDialog* (Figura 4.10) para que el usuario elija la geometría visual inicial. Tras la selección, se cargan sus vértices y caras, se construye el KDTree asociado y se carga el mapeo de índices XL-VIS correspondiente a esta geometría.

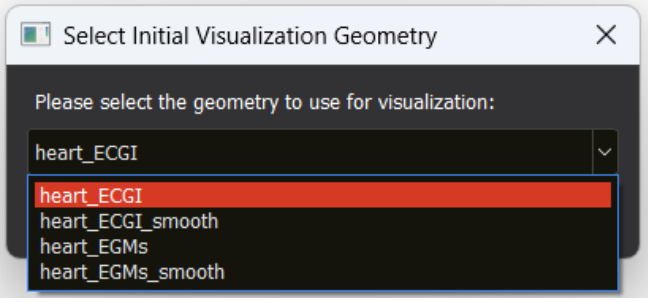


Figura 4.9 Diálogo de selección de la geometría visual inicial.

Una vez recogida esta información, si se eligió la generación vía MATLAB, se lanza *MatlabGenerationWorker* para preparar los archivos APD_curve.txt e initial_conditions.txt. Al finalizar este (o si los archivos se proveyeron directamente), el sistema invoca *AutomataManager.create_simulation* con todas las rutas y parámetros necesarios. El primer frame de datos se recupera de la DLL y se representa automáticamente sobre la malla visual seleccionada.

4.4.2 Reproducción, navegación temporal y sincronización visual

Tras la inicialización, el usuario puede reproducir la simulación mediante los controles del *BottomControlPanel.* Al activar la reproducción:

- Control de Reproducción: Al pulsar el botón "Play", la simulación comienza a animarse en la vista 3D. Internamente, un temporizador solicita y muestra nuevos fotogramas a la velocidad seleccionada por el usuario. Si la aplicación necesita generar más datos para continuar la

animación, solicita automáticamente nuevos lotes de cálculo al *SimulationWorker* para mantener el flujo.

- Navegación Manual: El usuario puede pausar la animación en cualquier momento, avanzar o retroceder frame a frame utilizando los botones dedicados, o saltar directamente a un instante específico mediante el slider de tiempo.
- Retroalimentación Visual: En todo momento, *MainWindow* asegura que la visualización 3D (colores de la malla que representan el potencial de acción) y los indicadores de tiempo (índice de frame y tiempo simulado en milisegundos) se actualicen para reflejar el estado actual de la simulación.

4.4.3 Definición y aplicación de estrategias de Ablación

Durante la reproducción o sobre un frame detenido, el usuario puede activar el modo de dibujo para definir lesiones de ablación:

1. **Activación del modo edición:** Se activa el modo de ablación pulsando el botón con el icono de un pincel en el BottomControlPanel o mediante el atajo de teclado 'E'.
2. **Dibujo interactivo:** Con el modo edición activo, el usuario puede hacer clic derecho y arrastrar el ratón sobre la geometría 3D para "pintar" las áreas de ablación (o borrarlas, si selecciona la herramienta de borrado). El tamaño del pincel es ajustable. Los cambios se visualizan inmediatamente en la malla como una preparación ("staging") de la lesión. Puede verse un ejemplo de esto en la Figura 4.11.
3. **Carga y aplicación de máscaras externas:** Alternativamente, el usuario puede cargar máscaras de ablación predefinidas (desde archivos .txt o .mat) seleccionándolas en el *RightControlPanel*. Estas máscaras se pueden añadir al área de preparación actual o reemplazarla. La vista 3D también se actualiza para mostrar el efecto combinado de estas máscaras preparadas.
4. **Aplicación a la simulación:** Una vez satisfecho con las lesiones preparadas, el usuario pulsa el botón "Apply to Sim". La aplicación entonces combina todas las lesiones preparadas (dibujo y máscaras externas) con cualquier ablación ya existente en la simulación, traduce esta lesión combinada al formato requerido por el motor de cálculo y la envía a la simulación. El estado de ablación de la simulación se actualiza permanentemente, y la visualización 3D refleja este cambio.
5. **Descarte de cambios:** Si el usuario no desea aplicar los cambios preparados, puede pulsar "Discard Changes". Esto elimina todas las lesiones que estaban en preparación, y la visualización vuelve a mostrar únicamente las ablaciones que ya estaban efectivas en la simulación.

La ablación aplicada se refleja visualmente de forma inmediata sobre la malla, y queda registrada en con el índice temporal asociado, asegurando que la visualización de la ablación se mantenga sincronizada al navegar por fotogramas pasados o al reproducir la simulación desde el inicio.

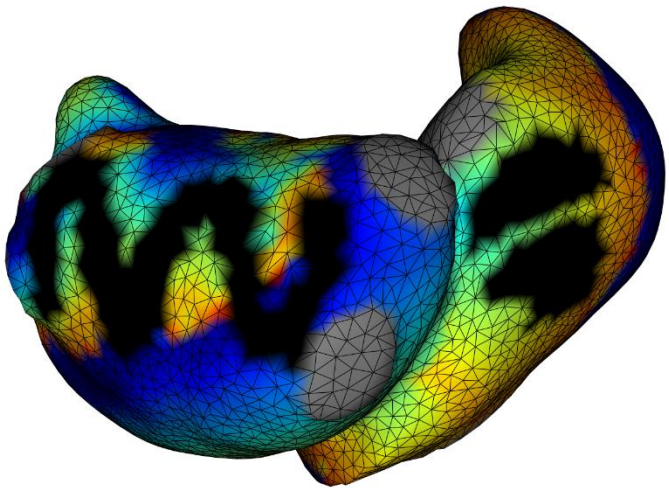


Figura 4.11 Muestra de la herramienta de dibujo interactivo. Ejemplo de lesiones virtuales (negro) trazadas directamente sobre la geometría con el modo wireframe activado para mostrar la malla triangular y como la ablación afecta a nodos individuales. Los trazos están hechos de forma aleatoria como ejemplo, no representan ninguna estrategia de ablación real.

Es importante destacar que el modelo actual de la DLL no incluye una representación explícita del nodo sinusal ni mecanismos para la generación de un ritmo sinusal espontáneo. Por lo tanto, cuando la actividad fibrilatoria cesa como resultado de una estrategia de ablación, el modelo evoluciona hacia un estado de reposo eléctrica, en lugar de revertir a un ritmo sinusal fisiológico

4.4.4 Cálculo y visualización de mapas funcionales

Cuando el usuario considera que un segmento del historial de simulación es de interés para un análisis más detallado puede generar mapas de características.

1. **Selección del rango temporal:** Desde el menú "Tools", el usuario selecciona "Calculate Frequency Maps" (Figura 4.12). Se abre un diálogo (*FrameRangeDialog*) donde especifica el segmento del historial de simulación *(raw_history_storage*) que desea analizar (Figura 4.13). El diálogo asegura que el rango tenga una duración mínima adecuada para el análisis espectral.

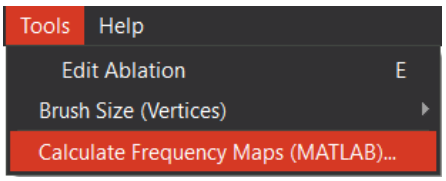


Figura 4.12 Barra de menú desplegada para mostrar como calcular los mapas de características.

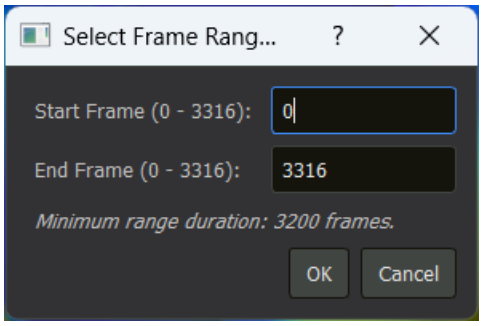


Figura 4.13 Diálogo de selección del rango de fotogramas del cual se quiere calcular los mapas.

2. **Extracción y mapeo de datos:** Los datos XL del segmento seleccionado se extraen y se mapean a la geometría VIS activa en ese momento.
3. **Ejecución del análisis en MATLAB:** Una vez confirmado el rango, la aplicación extrae los datos de simulación correspondientes, los prepara y los envía al *MatlabAnalysisWorker*, que ejecuta los scripts de MATLAB necesarios en segundo plano para calcular los diferentes mapas.
4. **Visualización de resultados:** Al finalizar el cálculo, *MainWindow* recibe los mapas. Para cada mapa, se crea una previsualización 3D interactiva (*MapPreviewWidget*) en una nueva pestaña dentro del panel *MapTabWidget*. El usuario puede explorar cada mapa en su preview (con su propia barra de color) y, si lo desea, seleccionar uno para mostrarlo superpuesto en la vista 3D principal para una inspección más detallada.

4.4.5 Escenarios de ablación

DIGICOR permite un flujo de trabajo iterativo para explorar el efecto de diferentes estrategias de ablación:

1. **Guardado de escenarios:** Desde el menú "Simulation", el usuario puede seleccionar "Save Ablation Scenario" (Figura 4.14). Esto guarda el estado completo de la simulación actual (incluyendo un rango de fotogramas del historial, la geometría, el mapeo y todas las ablaciones aplicadas hasta ese momento) con un nombre que especifica el usuario (Figura 4.15).

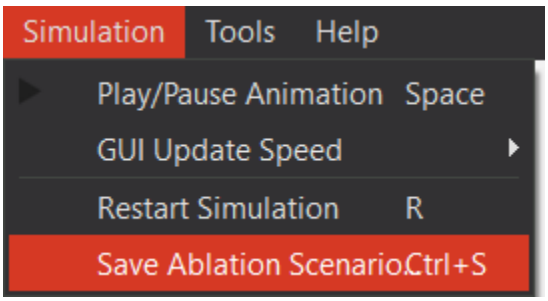


Figura 4.14 Barra de menú desplegada para mostrar cómo se accede al guardado de escenarios.

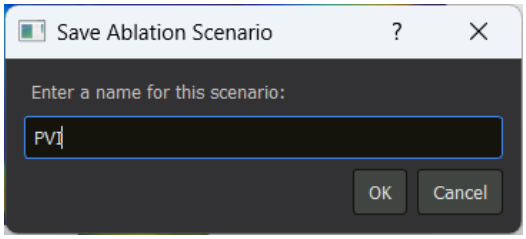


Figura 4.15 Diálogo para nombrar el escenario de ablación a guardar.

2. **Exploración individual:** Cada escenario guardado aparece en una nueva pestaña en el panel *ScenarioTabWidget*. Dentro de esta pestaña, un *ScenarioPreviewWidget* permite al usuario reproducir y navegar por ese escenario de forma independiente, con sus propios controles visuales.
3. **Comparación Simultánea:** Para comparar directamente el efecto de diferentes estrategias, el usuario puede activar el "Modo Comparación" desde el menú "View" (Figura 4.16). Luego, puede añadir los escenarios guardados que desee comparar a diferentes viewports en la vista 3D principal (usando la opción "Add Scenario to Comparison" del menú "View" como se puede ver también en la Figura 4.16). La aplicación muestra cada escenario en su propio viewport, permitiendo la reproducción sincronizada o individual y la observación lado a lado de sus dinámicas eléctricas.

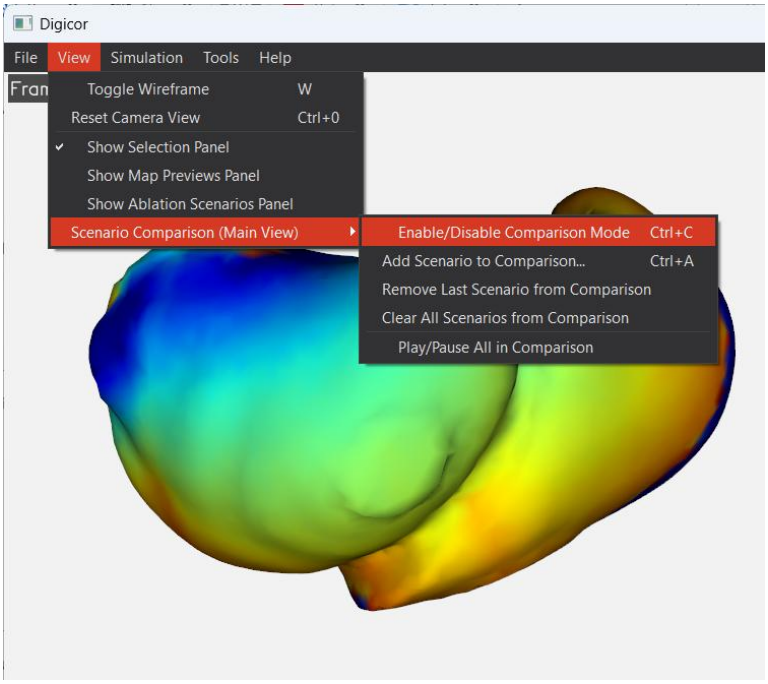


Figura 4.16 Interfaz mostrando la barra de menú desplegada para acceder a la activación del modo de comparación de escenarios. A su vez, se muestra como añadir un escenario o eliminarlo, limpiar todos los escenarios y poner en play/pause todos los escenarios simultáneamente.

Este flujo de trabajo permite al usuario evaluar cualitativamente las consecuencias de diversas estrategias de ablación, ayudando a identificar los enfoques más prometedores para la supresión de la actividad fibrilatoria.

5. Resultados

Este capítulo presenta los resultados obtenidos con la plataforma DIGICOR, dividiéndose en dos secciones principales. Primero, se describirán los resultados técnicos relacionados con la implementación y funcionalidad de la propia aplicación. Posteriormente, se detallarán los resultados experimentales obtenidos a través de la simulación de diferentes escenarios electrofisiológicos y la aplicación de estrategias de ablación, demostrando la utilidad práctica de DIGICOR.

5.1 Resultados técnicos

El desarrollo de DIGICOR ha culminado en una aplicación de escritorio funcional que integra simulación, visualización y análisis para el estudio de la FA. A continuación, se describen los principales resultados técnicos y funcionales logrados.

La **interfaz gráfica de usuario (GUI)**, construida con PyQt5, proporciona un entorno de trabajo organizado y flexible. El acceso a todas las funcionalidades es posible mediante menús, también existe una barra de estado informativa y un sistema de paneles acoplables. El panel central de visualización 3D permite al usuario alternar fluidamente entre un modo de simulación individual y un modo de comparación de múltiples escenarios, ambos renderizados mediante la biblioteca Vedo.

Los paneles de control (*RightControlPanel* y *BottomControlPanel*) ofrecen acceso intuitivo a las operaciones esenciales: El *RightControlPanel* facilita la carga y selección de múltiples geometrías auriculares actualizando dinámicamente la vista principal. También permite la gestión de máscaras de ablación externas, incluyendo su aplicación con modos de "reemplazo" o "adición" sobre las capas de preparación. El *BottomControlPanel* centraliza los controles de reproducción de la simulación (play/pausa, navegación por fotogramas, ajuste de velocidad) y las herramientas de ablación interactiva, como la activación del modo edición, la selección de la herramienta de dibujo/borrado y el ajuste del tamaño del pincel. Desde este panel también se gestiona la aplicación final de las ablaciones preparadas a la simulación o su descarte.

La **visualización 3D dinámica**, gestionada por *RenderManager*, representa eficientemente la propagación del potencial de acción mediante el coloreado en tiempo real de los vértices de la malla. Las lesiones de ablación, tanto las efectivas en la simulación como las que se encuentran en preparación, se muestran claramente sobre el modelo. La plataforma también soporta la superposición de mapas funcionales y texto informativo en la escena, así como la activación u ocultación del mallado de la geometría. La interacción 3D para la ablación, mediante *MyInteractorStyle* y *DrawingTool*, permite al usuario definir lesiones directamente sobre la malla, con retroalimentación visual inmediata de los cambios en las capas de preparación.

La **gestión de datos y la ejecución de tareas computacionales** se han diseñado para mantener la capacidad de respuesta de la GUI. La simulación del autómata celular (*SimulationWorker*) y las tareas que involucran MATLAB para la generación de archivos de entrada o el cálculo de mapas se ejecutan en hilos separados. La comunicación asíncrona mediante señales y slots de PyQt5 asegura que la interfaz permanezca activa, aunque se ha observado que la inicialización del motor MATLAB y la transferencia de datos voluminosos pueden introducir latencias en la disponibilidad de los resultados de estas tareas específicas.

Para la *gestión de datos y la animación*, se implementó un sistema sofisticado que equilibra el uso de memoria y la fluidez, basado en una arquitectura de caché predictiva. Los datos crudos del potencial de acción (formato XL) se almacenan secuencialmente en un repositorio en memoria

(raw_history_storage), que constituye el historial completo de la simulación y permite la navegación manual instantánea mediante el slider de tiempo.

El análisis cuantitativo del rendimiento de este sistema reveló dos regímenes de operación. En el primer régimen, durante la generación de datos en vivo, se constató que el rendimiento está limitado por la tasa de producción del backend. Las mediciones indicaron que esta tasa máxima es variable, oscilando típicamente entre 30 y 45 fotogramas por segundo (FPS), dependiendo directamente de la complejidad de la dinámica eléctrica simulada. Este hallazgo es coherente, ya que el "cuello de botella" computacional reside en el motor de la DLL.

En el segundo régimen, **al reproducir datos ya existentes en el historial**, el sistema exhibe un rendimiento significativamente superior, limitado únicamente por la capacidad de renderizado del frontend. En este escenario, la aplicación puede alcanzar tasas de visualización más altas. Se implementó una lógica de "marcapasos" en el *gui_update_timer* para que la velocidad de reproducción se ajuste con alta fidelidad a la solicitada por el usuario. Por ejemplo, al solicitar 60 FPS, se midieron tasas reales sostenidas de entre 50 y 60 FPS. Esto demuestra que, una vez que los datos han sido generados, el usuario puede analizarlos y reproducirlos a una velocidad muy superior a la que fue posible durante la simulación en vivo, facilitando una revisión rápida y eficiente de segmentos largos del historial.

En conclusión, la validación técnica confirma que DIGICOR ha logrado un diseño robusto que resuelve el desafío de la interactividad en la simulación científica. El sistema es capaz de gestionar la exigente carga computacional de la simulación mientras presenta su progreso de forma fluida, y ofrece al usuario un control de velocidad predecible y eficaz, diferenciando claramente entre la fase de generación de datos y la de análisis posterior.

Finalmente, la plataforma implementa la **gestión de escenarios de ablación** y **previsualización de mapas funcionales** mediante widgets dedicados (*ScenarioPreviewWidget, MapPreviewWidget*) alojados en pestañas dentro de paneles acoplables. Estos permiten el análisis detallado y la comparación de diferentes resultados y estrategias de forma independiente o integrada con la vista principal, facilitando un flujo de trabajo iterativo para la exploración de intervenciones.

En general, la aplicación demostró ser estable bajo las condiciones de prueba realizadas, aunque se identificaron puntos de mejora en la fluidez durante ciertas transiciones de estado complejas, como la carga inicial de la simulación.

5.2 Resultados experimentales

Para demostrar la aplicabilidad y el potencial de DIGICOR como herramienta de apoyo en la planificación de procedimientos de ablación para la FA, se llevaron a cabo una serie de simulaciones computacionales utilizando el modelo de autómata celular integrado. Las simulaciones se realizaron sobre una geometría auricular realista obtenida mediante ECGI imageless y procesada para su uso en la plataforma. Los parámetros fisiológicos base se ajustaron para simular tejido auricular sano, y se modificaron localmente para inducir y mantener la FA en los casos correspondientes.

El objetivo de estos experimentos es ilustrar cómo DIGICOR puede guiar la definición de estrategias de ablación más dirigidas y potencialmente menos extensivas que los protocolos estándar. A continuación, se presentan dos casos de estudio representativos que demuestran esta capacidad.

5.5.1 Caso de estudio 1

Para el primer caso de estudio, se configuró un sustrato auricular para generar un patrón de FA mantenido por un mecanismo estable localizado en la aurícula izquierda. La Figura 5.1 muestra dos

fotogramas de esta simulación, donde se puede apreciar un frente de onda que emana de la región de la orejuela izquierda. Se observa cómo el frente de activación (representado por colores cálidos como rojo y amarillo) circula alrededor de un punto, manteniendo la actividad reentrante y contribuyendo a la perpetuación de la arritmia en esta área.

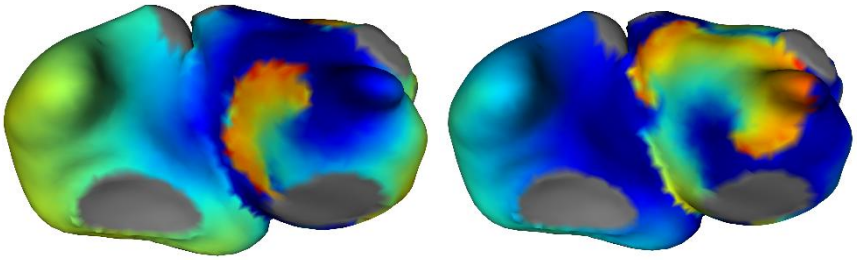


Figura 5.1 Simulación de FA: Secuencia de fotogramas mostrando el rotor en la orejuela auricular izquierda.

Para complementar la observación dinámica, se calcularon mapas funcionales sobre un segmento representativo de la simulación.

Para caracterizar objetivamente este comportamiento, se calcularon mapas funcionales a partir de un segmento representativo de la simulación. El mapa de estabilidad (Figura 5.2.a) identificó una región muy localizada, coincidente con la orejuela izquierda, con una alta persistencia de frecuencias elevadas (indicada en rojo). Esta localización fue confirmada por el mapa de frecuencia dominante media (Figura 5.2.b), que mostró un gradiente de frecuencia con el máximo en la misma área anatómica.

Finalmente, el mapa de histograma de patrones reentrantes (Figura 5.2.c) corroboró de forma inequívoca la existencia de un rotor estable y organizado, cuyo núcleo se situaba precisamente en la orejuela izquierda. La fuerte correspondencia entre los tres mapas funcionales permite identificar con alta confianza el mecanismo impulsor de la arritmia en este modelo.

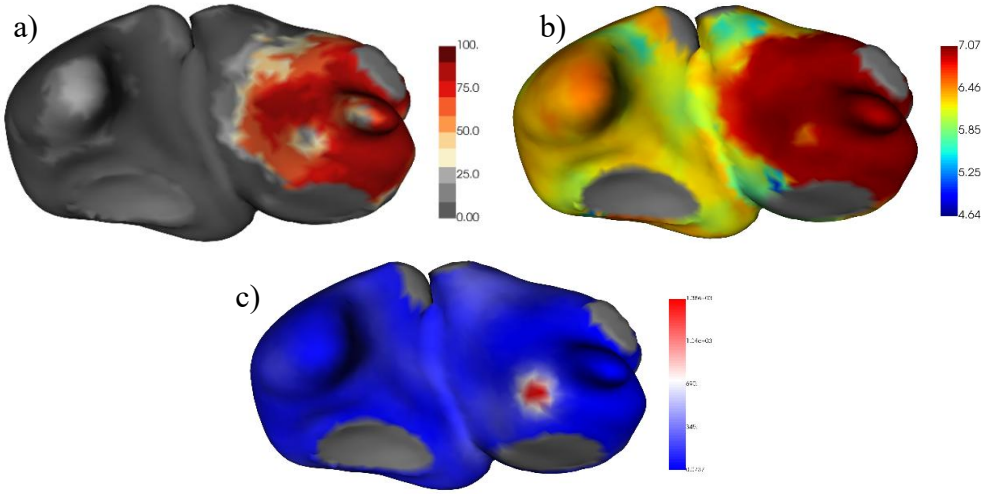


Figura 5.2 Mapas funcionales del escenario de FA con rotor en la orejuela izquierda. (a) Mapa de estabilidad, mostrando una alta persistencia (rojo) localizada en la orejuela. (b) Mapa de frecuencia dominante media, confirmando la orejuela como el foco de mayor frecuencia. (c) Mapa de histograma de patrones reentrantes, revelando un núcleo rotacional estable en la misma región.

Estrategia de ablación estándar de la práctica clínica

1. Aislamiento de Venas Pulmonares: Se aplicaron lesiones circunferenciales alrededor de las venas pulmonares. La Figura 5.3.a muestra la visualización de máscara de ablación (color negro) aplicada sobre la geometría. Se puede observar claramente que las regiones correspondientes a las venas pulmonares y el tejido circundante inmediato a las líneas de ablación permanecen eléctricamente inactivas (color azul oscuro), confirmando el éxito del aislamiento eléctrico virtual de estas estructuras. El resto de las imágenes, Figura 5.3.b y Figura 5.3.c muestran la persistencia de la FA.

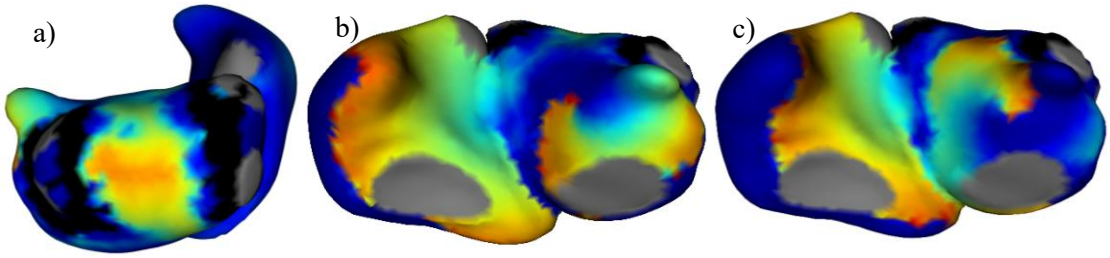


Figura 5.3 (a) Perspectiva de la pared posterior de la aurícula izquierda con PVI (negro). (b, c) Secuencia de dos fotogramas mostrando la persistencia de la FA tras aplicar PVI.

El cálculo de los mapas funcionales post-PVI confirmó esta observación. Los mapas de frecuencia dominante y de estabilidad mostraron valores nulos o mínimos en la zona aislada, pero los focos de alta frecuencia y estabilidad en la región de la orejuela izquierda permanecieron inalterados.

El análisis combinado de estos mapas y de la visualización de la simulación post-PVI refuerza la idea de que, aunque técnicamente exitoso en aislar las venas, la PVI no fue suficiente para terminar la FA en este modelo simulado y se necesita ablacionar otra región.

2. Ablación de Pared Posterior (PW): Siguiendo la progresión de una estrategia de ablación estándar, y dado que la FA persistió tras el PVI, el siguiente paso simulado fue la ablación de la pared posterior de la aurícula izquierda (Figura 5.4.a). A pesar de esta extensa modificación del sustrato en la aurícula izquierda, la observación de la dinámica eléctrica posterior reveló que la arritmia no fue terminada. Como se ilustra en el fotograma de la Figura 5.4.b, la actividad auricular caótica continuó, mantenida por el mismo rotor en la orejuela izquierda que se había identificado en la caracterización inicial.

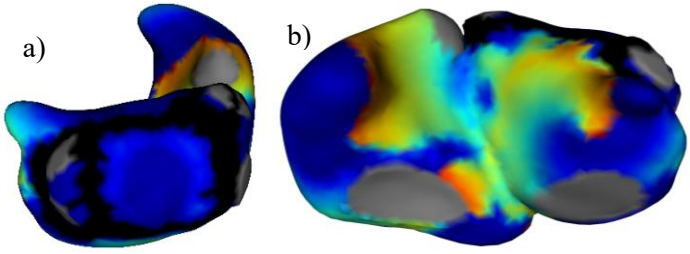


Figura 5.4 Persistencia de la FA tras la ablación combinada de PVI y PW. (a) Visualización del conjunto de lesiones aplicadas, que muestra el aislamiento completo de las venas pulmonares y la pared posterior. (b) Fotograma de la simulación post-ablación, donde se observa la persistencia de la FA.

3. Ablación de Líneas Adicionales

Considerando que la estrategia de ablación estándar extendida (PVI+PW) no logró terminar la arritmia, se procedió a simular una estrategia final dirigida por los hallazgos de los mapas

funcionales. El objetivo fue la fuente de la arritmia identificada de forma consistente: el rotor localizado en la orejuela izquierda (LAA).

Se aplicó una lesión de ablación virtual adicional para aislar eléctricamente la LAA. La Figura 5.5.a muestra un fotograma de la simulación inmediatamente antes de que el frente de onda del rotor se extinguiera por completo debido a la nueva lesión. Se puede observar cómo la actividad eléctrica compleja todavía está presente en la aurícula derecha y en la parte anterior de la aurícula izquierda, pero la fuente en la LAA ha sido contenida.

Inmediatamente después de la aplicación de la ablación de la LAA, la actividad eléctrica organizada cesó en todo el tejido auricular. La Figura 5.5.b muestra un fotograma posterior a la terminación, donde toda la aurícula se encuentra en un estado de reposo eléctrico (color azul oscuro), indicando el cese completo de la Fibrilación Auricular.

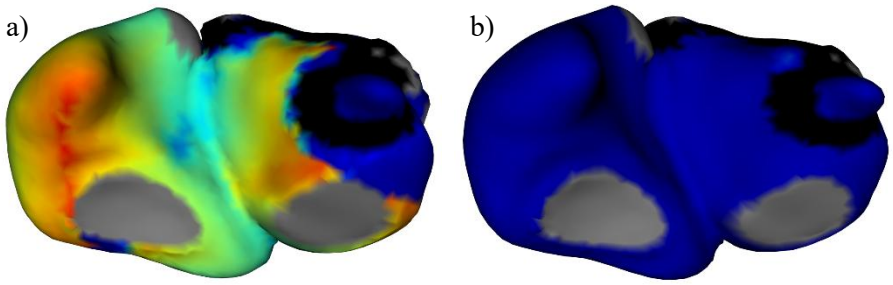


Figura 5.5 Terminación de la FA tras la ablación dirigida de la LAA. (a) Fotograma mostrando los últimos vestigios de actividad arrítmica justo antes de la terminación completa. (b) Estado de reposo eléctrico de toda la aurícula tras la ablación de la LAA, indicando el cese de la FA.

Dado que la FA terminó tras esta última etapa de ablación, no se calcularon mapas funcionales adicionales, ya que, su interpretación en ausencia de actividad arrítmica sostenida no aportaría información relevante.

La simulación secuencial de una estrategia de ablación estándar demostró que, para este modelo particular de FA, las lesiones en la aurícula izquierda (PVI y PW) no fueron suficientes por sí solas para terminar la arritmia. La terminación solo se logró tras la ablación adicional de la LAA. Aunque esta estrategia finalmente tuvo éxito, implicó una cantidad considerable de lesiones virtuales en ambas aurículas.

Estrategia de ablación dirigida por DIGICOR

Para demostrar el valor de una estrategia de ablación guiada por la caracterización funcional, se simuló un abordaje alternativo partiendo del mismo estado de FA basal. En este escenario, en lugar de seguir la secuencia anatómica estándar (PVI, PW), se aplicó directamente una única lesión virtual diseñada para aislar el rotor de la arritmia identificado previamente en la orejuela izquierda.

La Figura 5.6.a muestra un fotograma de la simulación justo antes de la terminación de la FA. Se puede observar que la actividad eléctrica compleja todavía persiste en el resto del tejido auricular, pero la fuente principal en la LAA ha sido contenida por la lesión de ablación.

Inmediatamente después, como se muestra en la Figura 5.6.b, la actividad arrítmica cesa por completo en toda la aurícula, que entra en un estado de reposo eléctrico. La terminación de la FA se logró con una única lesión, pequeña y precisa, sin necesidad de realizar el aislamiento extensivo de las venas pulmonares o la pared posterior.

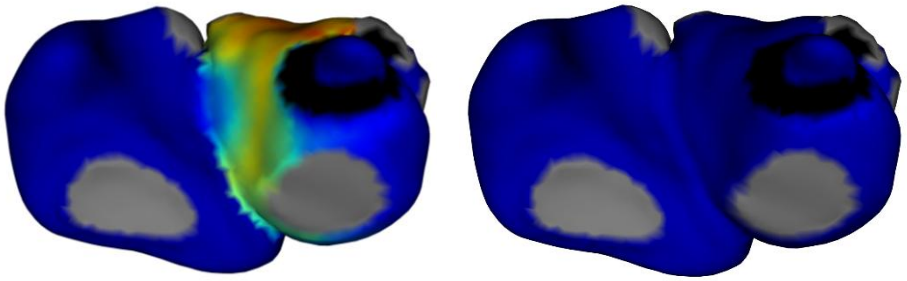


Figura 5.6 Terminación de la FA mediante una estrategia de ablación dirigida. (a) Fotograma mostrando la persistencia de la actividad eléctrica justo antes de la extinción, con la lesión de la LAA ya aplicada. (b) Cese completo de la arritmia, con toda la aurícula en estado de reposo eléctrico, tras la ablación única y selectiva de la LAA.

Este resultado es la principal conclusión del caso de estudio. Subraya el potencial de DIGICOR no solo para identificar los mecanismos de la FA, sino para guiar una terapia de ablación mucho más eficiente y conservadora. Al dirigir la ablación directamente a la fuente funcional, se podría, hipotéticamente, reducir significativamente la extensión de las lesiones, el tiempo del procedimiento y los riesgos asociados en comparación con los protocolos estándar, especialmente en casos de FA persistente donde los drivers no se localizan en las venas pulmonares.

5.5.2 Caso de estudio 2

El siguiente caso se trata de una FA caracterizada por la presencia de distintas reentradas en la pared lateral de la aurícula derecha y un rotor persistente en la orejuela auricular derecha, como se puede ver en la Figura 5.7.

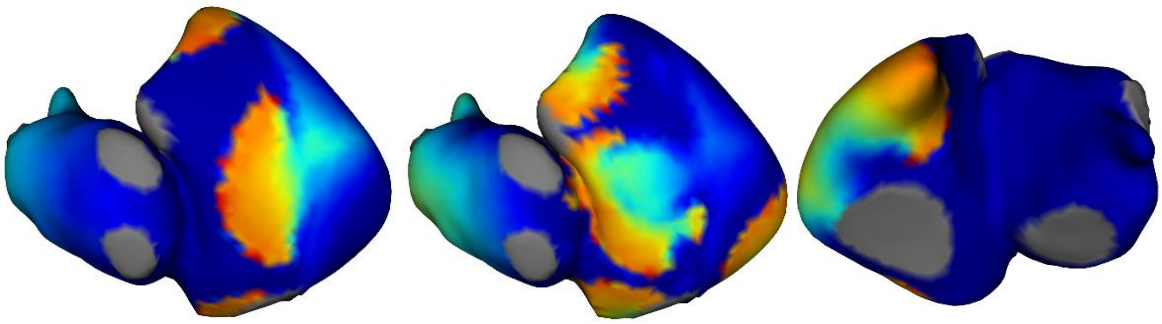


Figura 5.7 Simulación de FA: Secuencia de fotogramas mostrando varias reentradas en la pared lateral derecha y un rotor en la orejuela aurícula derecha.

Para identificar los mecanismos de mantenimiento de la arritmia simulada, se realizó un análisis funcional sobre la actividad eléctrica de la FA. Se obtuvieron los mapas funcionales correspondientes a esta FA, cuyos resultados se presentan en las Figura 5.8.

El el mapa de estabilidad (Figura 5.8.a) y el mapa de frecuencia dominante (Figura 5.8.b) revelaron de forma consistente la existencia de varias regiones con alta frecuencia en la pared lateral de la aurícula, además de una zona de máxima actividad en la orejuela auricular derecha.

El análisis fue corroborado por el mapa de patrones de reentrada (Figura 5.8.c), el cual identificó de manera inequívoca un rotor estable, localizado precisamente en la RAA, próximo a su base.

Estos hallazgos sugieren un mecanismo de FA donde un rotor principal en la RAA actúa como fuente de alta frecuencia, generando frentes de onda que se propagan de forma caótica por el resto del tejido auricular.

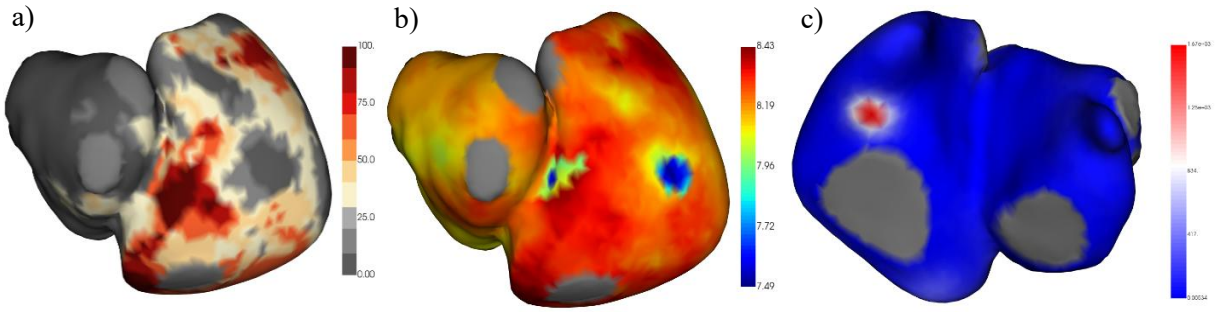


Figura 5.8 Mapas funcionales. (a) Mapa de estabilidad mostrando frecuencias altas en algunas zonas de la aurícula derecha. (b) Mapa de frecuencia dominante mostrando la pared lateral de la aurícula derecha con las frecuencias más altas de media. (c) Mapa de patrones de reentrada mostrando el rotor en la RAA.

Basándose en los resultados del análisis funcional, se diseñó y ejecutó una estrategia de ablación virtual por pasos. Primero, se aplicó una línea de ablación estándar en el istmo cavo-tricuspídeo y la crista terminalis. La Figura 5.9.a muestra que esta lesión fue insuficiente para terminar con la FA.

A continuación, se realizó una segunda ablación dirigida a la diana identificada por los mapas: se aisló eléctricamente la RAA. Esta intervención sí fue efectiva. La Figura 5.9.b captura el momento en que los últimos frentes de onda se extinguen al colisionar con las líneas de ablación. Finalmente, la Figura 5.9.c muestra el estado de reposo de todo el tejido auricular, confirmando la terminación completa de la FA.

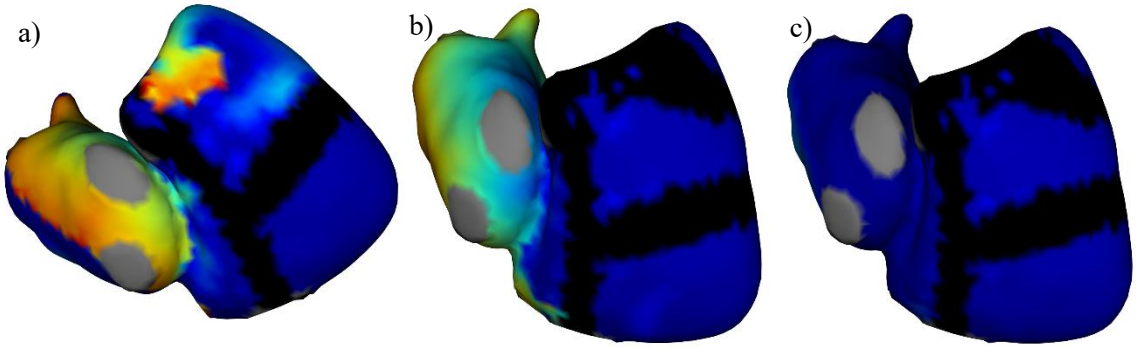


Figura 5.9 Estrategia de ablación virtual por pasos. (a) La primera ablación (ICT + Crista + RAA) no termina la FA. (b, c) La adición de una segunda línea de ablación para aislar la RAA, identificada como driver, provoca la terminación de la arritmia.

De nuevo, se observa cómo con una estrategia de ablación guiada por simulación se puede terminar con la FA sin necesidad de realizar una ablación extensiva. Es de destacar que, en este caso, la

arritmia fue resuelta sin intervenir sobre la aurícula izquierda, que es la que suele ser objeto de mayor daño tisular en los procedimientos de ablación estándar. Este resultado subraya el potencial de los modelos computacionales para desarrollar estrategias de ablación más eficientes y específicas para cada paciente, minimizando el daño innecesario al tejido sano.

Conclusión de los casos de estudio

La comparación entre las estrategias de ablación estándar y las estrategias dirigidas por DIGICOR en los casos presentados es reveladora. Al utilizar la información proporcionada por la visualización dinámica y los mapas funcionales, fue posible diseñar y simular virtualmente intervenciones mucho más conservadoras que lograron la terminación de la arritmia, en contraste con los extensos conjuntos de lesiones requeridos por los protocolos anatómicos estándar.

Estos escenarios subrayan el potencial de DIGICOR como herramienta de planificación preoperatoria. La plataforma permite no solo caracterizar los mecanismos específicos de la FA de un paciente simulado, sino también explorar y anticipar la eficacia de diferentes estrategias de ablación. En los casos estudiados, DIGICOR habría ayudado a evitar lesiones extensas e innecesarias, dirigiendo la terapia de forma precisa hacia los mecanismos críticos subyacentes. Este enfoque guiado por la simulación tiene el potencial de traducirse en procedimientos clínicos más cortos, seguros y eficaces para el paciente, demostrando el valor de la plataforma como un sistema de apoyo a la decisión clínica.

6. Pliego de condiciones

6.1 Presupuesto

Este apartado presenta una estimación aproximada de los recursos humanos y técnicos empleados en el desarrollo del Trabajo de Fin de Grado. El proyecto se ha llevado a cabo en el marco de unas prácticas remuneradas, lo que implica la existencia de un coste económico asociado. Se recoge una valoración general de la dedicación personal y de los medios materiales utilizados.

6.1.1 Costes de personal

Para la estimación de los costes de personal vinculados al desarrollo de este Trabajo de Fin de Grado, se han considerado tres figuras clave: la estudiante (autora del proyecto), el tutor académico y el director experimental.

En el caso del personal docente (tutor y director), se ha tomado como referencia el perfil del Personal Docente e Investigador (PDI) de la Universitat Politècnica de València, con un cálculo de coste por hora basado en una dedicación anual de 1696 horas. Las horas indicadas reflejan una estimación del tiempo efectivo dedicado al proyecto. Para el **director experimental**, se ha estimado una dedicación de **30 horas**, considerando reuniones semanales, asesoramiento técnico y seguimiento del trabajo.

En cuanto a la estudiante, se ha incluido su coste real, ya que ha realizado el trabajo en el marco de unas prácticas remuneradas, con una retribución de **6 €/h** y una dedicación total de **300 horas**.

Tabla 6.1 Costes de personal

Nº	Descripción del recurso humano	Precio (€/h)	Horas (h)	Total (€)
1	Tutor académico	20.39,00	26	530.14,00
2	Director experimental	19.37,00	30	581.1,00
3	Estudiante de Ingeniería Física en prácticas	6,00	300	1800,00
			Total (€)	**2911.24,00 €**

6.1.2 Costes de materiales

El proyecto se ha desarrollado íntegramente con recursos personales y herramientas de software libre. La única amortización incluida corresponde al equipo informático principal utilizado durante el desarrollo del proyecto, ya que el software empleado ha sido de código abierto o proporcionado por la universidad.

Tabla 6.2 Costes de materiales

Nº	Descripción del material	Precio unitario (€/u)	Cantidad (unidades)	Total (€)
1	Portátil con procesador Intel Core i5 y 8 GB de RAM	668,00	1	668,00

2	Monitor externo	129,00	1	129,00
3	Teclado y ratón (pack periféricos de oficina)	50,00	1	50,00
4	Python + librerías (PyQt5, VTK, etc.)	0	1	0
5	Visual Studio Code	0	1	0
6	Microsoft 365 (licencia UPV)	0	1	0
			Total	**847,00 €**

6.1.3 Presupuesto total estimado

A continuación, se presenta el resumen global de los costes derivados del proyecto, resultantes de la suma de los recursos humanos y materiales descritos en los apartados anteriores.

Tabla 6.3 Coste total

Nº	**Descripción**	**Precio (€)**
1	Coste de personal	2911.24,00
2	Coste de material	847,00
	Total	**3758.24,00 €**

6.2 Recursos técnicos

Los recursos tecnológicos empleados durante el desarrollo del proyecto, incluyendo bibliotecas de simulación, entornos de programación, herramientas de visualización y plataformas de análisis, han sido descritos previamente en el capítulo de Metodología. En el presente apartado no se listan de nuevo, ya que su uso no ha implicado un coste económico adicional, al tratarse de herramientas de software libre o con licencias institucionales.

6.3 Materiales y equipos

El proyecto se ha desarrollado utilizando equipamiento informático proporcionado por la entidad colaboradora, incluyendo un portátil de altas prestaciones, monitor externo y periféricos básicos. Estos elementos han sido considerados en el presupuesto como recursos materiales asignados al trabajo, con una valoración estimada basada en su uso durante el periodo de desarrollo.

7. Conclusiones

El presente Trabajo de Fin de Grado ha culminado con el desarrollo exitoso de **DIGICOR**, una plataforma de software pionera, funcional e interactiva para la simulación personalizada de la fibrilación auricular y la evaluación virtual de estrategias de ablación. La plataforma integra con éxito un motor de simulación de alto rendimiento, herramientas de análisis funcional y una interfaz gráfica de usuario intuitiva, demostrando ser una herramienta con un alto potencial para la planificación de procedimientos clínicos y la investigación en el campo de las arritmias cardíacas. A continuación, se detallan las conclusiones específicas alcanzadas en relación con los objetivos planteados.

1. Integración del motor de simulación y los módulos de análisis:

Se ha logrado la completa integración del motor de simulación *AutomataCUDAv1.dll* y de los scripts de análisis de MATLAB en una única arquitectura de software controlada por Python. Esto se consiguió mediante el uso de la librería *ctypes* para la comunicación con la DLL y el MATLAB Engine API, estableciendo un flujo de datos robusto que sienta las bases de toda la funcionalidad de la plataforma. Este primer hito fue crucial para transformar componentes aislados en un sistema cohesionado y funcional.

2. Desarrollo del frontend:

Se ha diseñado e implementado una interfaz gráfica de usuario bajo el framework PyQt5, que ofrece una experiencia de usuario fluida e intuitiva. La arquitectura de paneles acoplables, el renderizado 3D mediante Vedo/VTK y las herramientas interactivas, como el pincel de ablación, permiten al usuario no solo visualizar la dinámica eléctrica, sino también interactuar directamente con el modelo geométrico en tiempo real, facilitando la definición de lesiones de forma precisa y visual.

3. Desarrollo del backend:

Se ha construido un backend con una arquitectura multihilo que gestiona de forma asíncrona las tareas computacionalmente intensivas. La lógica de control implementada orquesta eficazmente la ejecución de la simulación en un hilo secundario, mientras la interfaz gráfica permanece completamente receptiva. Este desacoplamiento entre el cálculo y la presentación visual es una de las claves del rendimiento de DIGICOR y permite una experiencia de usuario sin interrupciones, incluso durante la generación de datos.

4. Integración y pruebas del sistema:

La plataforma ha sido sometida a un riguroso proceso de pruebas funcionales que han validado la correcta integración de todos sus módulos. La comunicación asíncrona mediante el sistema de señales y slots de PyQt5 ha demostrado ser estable, asegurando la coherencia entre los datos generados por el backend y su representación visual en el frontend. Las pruebas de consistencia y los casos de uso simulados confirman la fiabilidad y robustez de la aplicación.

5. Evaluación de estrategias de ablación:

Se ha implementado con éxito la funcionalidad para la evaluación y comparación de múltiples escenarios de ablación. Los casos de estudio presentados en este trabajo demuestran que DIGICOR puede identificar los mecanismos impulsores de la FA y guiar estrategias de ablación más dirigidas y menos extensivas que los protocolos estándar. La capacidad de comparar visualmente diferentes abordajes terapéuticos lado a lado se erige como una potente herramienta para la toma de decisiones clínicas.

6. Optimización y mejora del sistema:

Se han realizado mejoras significativas en la usabilidad y el rendimiento de la aplicación, optimizando la velocidad de respuesta de la interfaz y la visualización. El análisis crítico realizado ha permitido identificar las fortalezas de la plataforma, como su interactividad y modularidad, así como áreas de mejora que se detallan en las secciones siguientes.

7. Documentación de la plataforma:

Se ha elaborado la documentación técnica fundamental de la solución, incluyendo diagramas de arquitectura y de flujo de datos que detallan la estructura interna y la lógica de la aplicación. Asimismo, se ha creado una guía de usuario básica en forma de atajos de teclado (Anexo A) y un diagrama de experiencia de usuario (Anexo B), que facilitan la comprensión y el uso de la plataforma, asegurando la transferibilidad del conocimiento generado.

Limitaciones de la aplicación

Pese a sus méritos, DIGICOR todavía enfrenta importantes desafíos que deben ser superados para garantizar su adopción clínica generalizada. En primer lugar, la aplicación no ha sido aún validada sistemáticamente en entornos hospitalarios, y su evaluación se ha limitado a pruebas de concepto o entornos controlados. Esto implica que su eficacia predictiva en condiciones reales sigue siendo una hipótesis por confirmar.

En segundo lugar, la calidad de las simulaciones depende críticamente de la precisión de los datos de entrada. El pipeline actual requiere geometrías segmentadas y datos clínicos precisos, lo cual puede verse afectado por errores en la adquisición o el preprocesamiento. Esto limita, de momento, su uso sin supervisión experta o en centros sin personal técnico especializado.

Desde el punto de vista del desarrollo, la versión actual carece de algunas funcionalidades clave para una experiencia de usuario robusta: no dispone de barra de progreso, historial de ablaciones, ni análisis automático de patrones, lo cual reduce su funcionalidad en escenarios clínicos complejos. No obstante, se ha desarrollado documentación detallada para usuarios finales o desarrolladores, lo que permite la transferencia y escalabilidad del software.

Una limitación del motor de simulación actual es la ausencia de un modelo del nodo sinusal y de los mecanismos de generación de ritmo fisiológico. Esto implica que, si bien la plataforma puede simular eficazmente la perpetuación y terminación de la FA, el estado post-terminación es de reposo eléctrico en lugar de un retorno al ritmo sinusal. De hecho, estamos ya estudiando su implementación para un futuro próximo.

La arquitectura, aunque prometedora, todavía no contempla componentes como simulación de mecánica cardíaca, análisis hemodinámico o integración directa con sistemas hospitalarios, lo que restringe su alcance a nivel funcional. Finalmente, si bien se ha mejorado la compatibilidad con Python 3.9+, aún no se ha migrado a tecnologías más recientes como PyQt6, que podrían facilitar la compatibilidad con sistemas modernos y pantallas HiDPI.

Perspectivas de crecimiento futuro

El desarrollo de DIGICOR se encuentra en un punto de inflexión prometedor, habiendo establecido una base funcional sólida para la simulación interactiva de la fibrilación auricular. El plan de mejoras elaborado detalla una hoja de ruta clara que incluye, como prioridades iniciales, la estabilización del código y la optimización del rendimiento, enfocándose en aspectos como la mejora de la latencia en la interacción con MATLAB y la mitigación de interrupciones momentáneas de la interfaz durante transiciones de estado complejas. Paralelamente, se contempla la ampliación de funcionalidades,

como la posibilidad de visualizar la geometría de simulación de alta resolución (XL), y una validación clínica rigurosa en entornos hospitalarios.

En fases sucesivas, se prevé la implementación de un sistema de eventos y plugins para extender la modularidad, la integración con APIs REST para la interoperabilidad con otros sistemas, y capacidades avanzadas como la simulación por lotes y el análisis de sensibilidad. Estas mejoras no solo expanden las capacidades técnicas del software, sino que también lo acercan progresivamente a un uso clínico real y robusto.

La incorporación de funcionalidades de análisis más sofisticadas, como la detección y caracterización automática de reentradas, la exportación de resultados de simulación en formato de vídeo para una mejor comunicación, o incluso el soporte experimental para tecnologías de realidad virtual, permitirán nuevos modos de interacción y análisis, beneficiando tanto a clínicos como a investigadores. Además, se planea la inclusión de nuevos tipos de mapas, como mapas de activación local, representación de vectores de propagación y la capacidad de realizar cortes internos en el modelo 3D, que enriquecerán el entorno de simulación con mayor profundidad espacial y funcional.

Desde una perspectiva estratégica, DIGICOR tiene el potencial de evolucionar hacia una plataforma integral para el estudio y tratamiento personalizado de arritmias cardíacas. Su arquitectura modular, diseñada para la eficiencia y su compatibilidad con la aceleración por GPU, lo posicionan favorablemente para su escalado y adaptación. A medio plazo, se espera avanzar en su validación en centros de referencia y explorar su integración con sistemas de historiales clínicos electrónicos, lo que permitiría una experiencia completamente integrada en el flujo asistencial del paciente.

A largo plazo, la aplicación podría extenderse al estudio de otras arritmias auriculares. Asimismo, podría beneficiarse significativamente de la incorporación de técnicas de inteligencia artificial para la identificación automatizada de regiones diana para la ablación, la predicción de la respuesta al tratamiento o la estimación del riesgo de recurrencias. Todo ello sitúa a DIGICOR como una plataforma con un gran potencial para transformar la planificación del tratamiento de la FA y contribuir al paradigma emergente de la medicina personalizada basada en simulación.

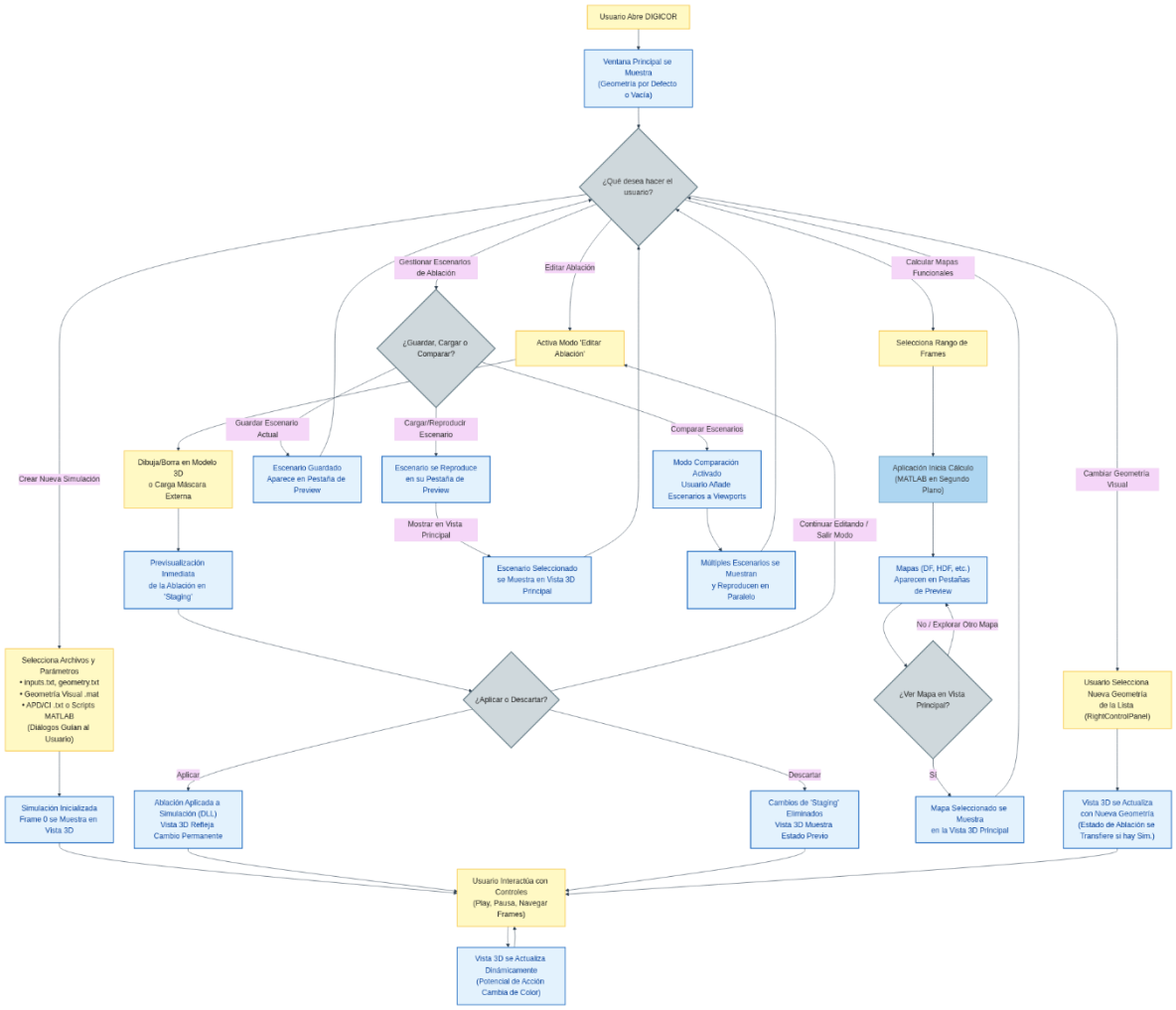
Anexo

Anexo A

Key	Action
Space	Play/Pause GUI animation (single simulation view)
E	Toggle "Edit" mode
W	Toggle wireframe/solid view
R	Restart simulation (clears ablations and data history)
Ctrl + M	Create Simulation (Generate via MATLAB)
Ctrl + T	Create Simulation (from TXT Files)
Ctrl + D	Destroy current simulation instance
Ctrl + S	Save Ablation Scenario
← / →	Previous / Next frame (single simulation view)
Shift + ← / **Shift** + →	Go back / forward 10 frames (single simulation view)
Ctrl + 0	Reset camera view
Ctrl + C	Enable/Disable Scenario Comparison Mode
Ctrl + A	Add Scenario to Comparison...
Ctrl + Q	Exit application
F1	Show this help

Figura A Atajos del teclado. Se puede acceder desde Help en la barra de menú.

Anexo B



B. Diagrama del flujo de experiencia del usuario.

Referencias

[1] Hindricks, G.; Potpara, T.; Dagres, N.; Bax, J. J.; Boriani, G.; Dan, G. A. *et al.*, "2020 ESC Guidelines for the diagnosis and management of atrial fibrillation developed in collaboration with the European Association for Cardio-Thoracic Surgery (EACTS)," Eur. Heart J., vol. 42, pp. 373–498, 2021. doi: 10.1093/eurheartj/ehaa612.

[2] Odutayo, A.; Wong, C. X.; Hsiao, A. J.; Hopewell, S.; Altman, D. G.; Emdin, C. A. et al., "Atrial fibrillation and risks of cardiovascular disease, renal disease, and death: systematic review and meta-analysis," BMJ, vol. 354, p. i4482, Sept. 2016. doi: 10.1136/bmj.i4482.

[3] Yamamoto, C. and Trayanova, N. A., "Atrial fibrillation: Insights from animal models, computational modeling, and clinical studies," EBioMedicine, vol. 85, p. 104310, Nov. 2022. doi: 10.1016/j.ebiom.2022.104310.

[4] Chugh, S. S. et al., "Worldwide epidemiology of atrial fibrillation: a Global Burden of Disease 2010 Study," Circulation, vol. 129, no. 8, pp. 837–847, Feb. 2014. doi: 10.1161/CIRCULATIONAHA.113.005119.

[5] Krijthe, B. P. et al., "Projections on the number of individuals with atrial fibrillation in the European Union, from 2000 to 2060," European Heart Journal, vol. 34, no. 35, pp. 2746–2751, Sept. 2013. doi: 10.1093/eurheartj/eht280.

[6] Lau, D. H.; Nattel, S.; Kalman, J. M. and Sanders, P., "Modifiable risk factors and atrial fibrillation," Circulation, vol. 136, no. 6, pp. 583–596, Aug. 2017. doi: 10.1161/CIRCULATIONAHA.116.023163.

[7] Patel, N. J.; Deshmukh, A.; Pant, S.; Singh, V.; Patel, N.; Arora, S. et al., "Contemporary trends of hospitalization for atrial fibrillation in the United States, 2000 through 2010: implications for healthcare planning," Circulation, vol. 129, no. 23, pp. 2371–2379, Jun. 2014. doi: 10.1161/CIRCULATIONAHA.114.008201.

[8] Camm, A. J.; Lip, G. Y. H.; De Caterina, R. et al., "2012 focused update of the ESC Guidelines for the management of atrial fibrillation," European Heart Journal, vol. 33, no. 21, pp. 2719–2747, Nov. 2012. doi: 10.1093/eurheartj/ehs253.

[9] Chamberlain, A. M.; Weston, S. A.; Yao, X.; Roger, V. L.; Noseworthy, P. A.; Gersh, B. J.; Shen, W. K.; Manemann, S. M.; Bailey, K. R.; Killian, J. M.; Friedman, P. A.; Asirvatham, S. J. and Packer, D. L., "Atrial fibrillation and mortality in heart failure: a community study," Circulation: Heart Failure, vol. 4, no. 6, pp. 740–746, Nov. 2011. doi: 10.1161/CIRCHEARTFAILURE.111.962688.

[10] Ball, J.; McMurray, J. J. V.; Semple, S. J. G.; Thompson, D. R.; Roger, S. P. S.; Wright, D. J.; Frenneaux, M. P.; Thompson, L. J.; Mills, P. R. and Cleland, J. G. F., "Atrial fibrillation: Profile and burden of an evolving epidemic in the 21st century," International Journal of Cardiology, vol. 167, no. 5, pp. 1807–1824, Sept. 2013. doi: 10.1016/j.ijcard.2012.12.093.

[11] Haissaguerre, M.; Jaïs, P.; Shah, D. C.; Takahashi, A.; Hocini, M.; Quiniou, G.; Garrigue, S.; Le Mouroux, A.; Le Métayer, P. and Clémenty, J., "Spontaneous initiation of atrial fibrillation by ectopic beats originating in the pulmonary veins," New England Journal of Medicine, vol. 339, no. 10, pp. 659–666, Sept. 1998. doi: 10.1056/NEJM199809033391003.

[12] Ramírez-Barrera, J. D.; Agudelo-Uribe, J. F.; Correa-Velásquez, R. and González-Rivera, E., "Fisiopatología de la fibrilación auricular," Revista Colombiana de Cardiología, vol. 23, no. S5, pp. 9–14, Nov. 2016. doi: 10.1016/j.rccar.2016.10.004.

[13] Moe, G. K. and Abildskov, J. A., "Atrial fibrillation as a self-sustaining arrhythmia independent of focal discharge," American Heart Journal, vol. 58, no. 1, pp. 59–70, Jul. 1959. doi: 10.1016/0002-8703(59)90274-1.

[14] García-Cosío, F., "¿Qué es y cómo se diagnostica la fibrilación auricular?," Revista Española de Cardiología, vol. 60, no. 2, pp. 93–96, Feb. 2007. doi: 10.1157/13099453.

[15] Jalife, J., "Rotors and spiral waves in atrial fibrillation," Journal of Cardiovascular Electrophysiology, vol. 14, no. 7, pp. 776–780, Jul. 2003. doi: 10.1046/j.1540-8167.2003.03136.x.

[16] Guillem, M. S., Climent, A. M., Rodrigo, M., Fernández Avilés, F., Atienza, F., y Berenfeld, O., "Presence and stability of rotors in atrial fibrillation: Evidence and therapeutic implications," Cardiovascular Research, vol. 109, no. 4, pp. 480–492, Mar. 2016. doi: 10.1093/cvr/cvw011.

[17] Allessie, M.; Ausma, J. and Schotten, U., "Electrical, contractile and structural remodeling during atrial fibrillation," Cardiovascular Research, vol. 54, no. 2, pp. 230–246, May 2002. doi: 10.1016/S0008-6363(02)00258-4.

[18] Mandapati, R.; Skanes, A.; Chen, J.; Berenfeld, O. and Jalife, J., "Stable microreentrant sources as a mechanism of atrial fibrillation in the isolated sheep heart," Circulation, vol. 101, no. 2, pp. 194–199, Jan. 2000. doi: 10.1161/01.CIR.101.2.194.

[19] Calkins, H.; Hindricks, G.; Cappato, R.; Kim, Y.-H.; Saad, E. B.; Aguinaga, L. et al., "2017 HRS/EHRA/ECAS/APHRS/SOLAECE expert consensus statement on catheter and surgical ablation of atrial fibrillation," Heart Rhythm, vol. 14, no. 10, pp. e275–e444, Oct. 2017. doi: 10.1016/j.hrthm.2017.05.012.

[20] Narayan, S. M. et al., "Treatment of atrial fibrillation by the ablation of localized sources: CONFIRM (Conventional Ablation for Atrial Fibrillation With or Without Focal Impulse and Rotor Modulation) trial," Journal of the American College of Cardiology, vol. 60, no. 7, pp. 628–636, Aug. 2012. doi: 10.1016/j.jacc.2012.05.022.

[21] Verma, A.; Jiang, C.-y.; Betts, T. R.; Chen, J.; Deisenhofer, I.; Mantovan, R. et al., "Approaches to catheter ablation for persistent atrial fibrillation," New England Journal of Medicine, vol. 372, no. 19, pp. 1812–1822, May. 2015. doi: 10.1056/NEJMoa1408288.

[22] Althoff, T. F. et al., "Standardized regionalization of the atria for 3D cardiac imaging, electroanatomical mapping and computational modeling: A multidisciplinary consensus of the PersonalizeAF consortium," Europace, vol. 26, no. S1, p. euae102.064, May. 2024. doi: 10.1093/europace/euae102.064.

[23] Fauchier, L. et al., "Beta-blockers or digoxin for atrial fibrillation and heart failure?," Cardiac Failure Review, vol. 2, no. 1, pp. 35–39, Apr. 2016. doi: 10.15420/cfr.2015:28:2.

[24] Kistler, P. M.; Chieng, D.; Sugumar, H.; Ling, L. H.; Segan, L.; Azzopardi, S.; Al-Kaisey, A.; Parameswaran, R.; Anderson, R. D.; Hawson, J. et al., "Effect of catheter ablation using pulmonary vein isolation with vs without posterior left atrial wall isolation on atrial arrhythmia recurrence in patients with persistent atrial fibrillation: The CAPLA randomized clinical trial,"

Journal of the American Medical Association, vol. 329, no. 2, pp. 127–135, Jan. 2023. doi: 10.1001/jama.2022.23722

[25] Romero, J. et al., "Electroanatomic mapping systems (CARTO/EnSite NavX) vs. conventional mapping for ablation procedures in a training program," Journal of Interventional Cardiac Electrophysiology: An International Journal of Arrhythmias and Pacing, vol. 45, no. 1, pp. 71–80, Feb. 2016. doi: 10.1007/s10840-015-0073-6.

[26] Molero Alabau, R., "Estimation of atrial electrical complexity during atrial fibrillation by solving the inverse problem of electrocardiography," Ph.D. thesis, Universitat Politècnica de València, Valencia, Spain, 2023.

[27] Taccardi, B.; Punske, B. B.; Lux, R. L.; MacLeod, R. S.; Ershler, P. R.; Dustman, T. J. and Vyhmeister, Y., "Useful lessons from body surface mapping," Journal of Cardiovascular Electrophysiology, vol. 9, no. 7, pp. 773–786, Jul. 1998. doi: 10.1111/j.1540-8167.1998.tb00965.x.

[28] Guillem Sanchez, M., "Activation patterns in atrial fibrillation: Contributions of body surface potential mapping," Ph.D. thesis, Universidad Politécnica de Valencia, Valencia, Spain, 2008.

[29] Molero, R. et al., "Robustness of imageless electrocardiographic imaging against uncertainty in atrial morphology and location," Journal of Electrocardiology, vol. 77, no. 1, pp. 58–61, Jan.-Feb. 2023. doi: 10.1016/j.jelectrocard.2022.12.007.

[30] Reventos-Presmanes, J.; Fambuena, C.; Cano-Cabañero, A.; Correas, M.; Hernández-Romero, I.; Herrero-Martin, C. et al., "Real-time cardiac mapping with a noninvasive imageless electrocardiographic imaging system," Journal of Visualized Experiments, no. 218, p. e67958, Apr. 2025. doi: 10.3791/67958.

[31] Guillem, M. S.; Climent, A. M.; Castells, F.; Husser, D.; Millet, J.; Arya, A.; Piorkowski, C. and Bollmann, A., "Noninvasive mapping of human atrial fibrillation," Journal of Cardiovascular Electrophysiology, vol. 20, no. 5, pp. 507–513, May. 2009. doi: 10.1111/j.1540-8167.2008.01356.x.

[32] MacLeod, R. and Buist, M., "The Forward Problem of Electrocardiography," in Comprehensive Electrocardiology, Macfarlane, P. W.; van Oosterom, A.; Pahlm, O.; Kligfield, P.; Janse, M. and Camm, J., Eds. London: Springer, pp. 247–298, 2010. doi: 10.1007/978-1-84882-046-3_8.

[33] Pullan, A. J.; Cheng, L. K. and Buist, M. L., "The Inverse Problem of Electrocardiology," in Mathematical Modeling of the Electrical Activity of the Heart, pp. 329–373, 2005. doi: 10.1007/978-1-84882-046-3_9

[34] Tikhonov, A. and Arsenin, V., Solution of Ill-posed Problems. New York, NY: John Wiley & Sons, 1977.

[35] Hansen, P. C., "Analysis of discrete ill-posed problems by means of the L-curve," SIAM Review, vol. 34, no. 4, pp. 561–580, Dec. 1992. doi: 10.1137/1034115.

[36] Ambellan, F. et al., "Statistical Shape Models: Understanding and Mastering Variation in Anatomy," in Advances in experimental medicine and biology vol. 1156, pp. 67-84, 2019. doi: 10.1007/978-3-030-19385-0_5.

[37] The openCARP consortium, "About openCARP," https://opencarp.org/about. [Online].

[38] ZMT Zurich MedTech AG, "Sim4Life: Computational Life Sciences Platform," https://sim4life.swiss/. [Online].

[39] King's College London, "OpenEP | Open-Source Electrophysiology Software," https://openep.io/. [Online].

[40] Oliphant, T. E., "Python for scientific computing," Computing in Science & Engineering, vol. 9, no. 3, pp. 10–20, May-June 2007. doi: 10.1109/MCSE.2007.58.

[41] Summerfield, M., Rapid GUI Programming with Python and Qt: The Definitive Guide to PyQt Programming. Pearson Education, 2015.

[42] Schroeder, W.; Martin, K. and Lorensen, W., The Visualization Toolkit, An Object-Oriented Approach To 3D Graphics, 4th ed. Clifton Park, NY: Kitware, Inc., 2006.

[43] vedo, "A python library for scientific analysis and visualization of 3D objects," https://vedo.embl.es/. [Online].

[44] Nickolls, J.; Buck, I.; Garland, M. and Skadron, K., "Scalable parallel programming with CUDA," ACM Queue, vol. 6, no. 2, pp. 40–53, Mar. 2008. doi: 10.1145/1365490.1365500.

[45] Greenberg, J. M. and Hastings, S. P., "Spatial patterns for discrete models of diffusion in excitable media," SIAM Journal on Applied Mathematics, vol. 34, no. 3, pp. 515–523, May. 1978. doi: 10.1137/0134040.

[46] Harris, C. R.; Millman, K. J.; van der Walt, S. J. et al., "Array programming with NumPy," Nature, vol. 585, no. 7825, pp. 357–362, Sept. 2020. doi: 10.1038/s41586-020-2649-2.

[47] Virtanen, P.; Gommers, R.; Oliphant, T. E. et al., "SciPy 1.0: Fundamental algorithms for scientific computing in Python," Nature Methods, vol. 17, no. 3, pp. 261–272, Mar. 2020. doi: 10.1038/s41592-019-0686-2.